IOWA STATE UNIVERSITY
Digital Repository

Graduate Theses and Dissertations

Iowa State University Capstones, Theses and Dissertations

2020

Semi-supervised learning algorithm to estimate mass flow from sparsely annotated images using a vision system (SLEM)

Muhammad Hamdan
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/etd

Recommended Citation

Hamdan, Muhammad, "Semi-supervised learning algorithm to estimate mass flow from sparsely annotated images using a vision system (SLEM)" (2020). *Graduate Theses and Dissertations*. 18321.
https://lib.dr.iastate.edu/etd/18321

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Graduate Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.

Semi-supervised learning algorithm to estimate mass flow from sparsely annotated images using a vision system (SLEM)

by

Muhammad K.A. Hamdan

A dissertation submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Co-majors: Electrical Engineering (Very Large Scale Integration)

Computer Engineering (Computing and Networking Systems)

Program of Study Committee:
Diane T. Rover, Major Professor
Matthew J. Darr
Chinmay Hegde
Philip H. Jones
Mani Mina

The student author, whose presentation of the scholarship herein was approved by the program of study committee, is solely responsible for the content of this dissertation. The Graduate College will ensure this dissertation is globally accessible and will not permit alterations after a degree is conferred.

Iowa State University

Ames, Iowa

2020

Copyright © Muhammad K.A. Hamdan, 2020. All rights reserved.

DEDICATION

I would like to dedicate this dissertation to my family for their love and support. I would also like to thank my friends for their advice, and good companionship.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

ACKNOWLEDGMENTS

I would like to take this opportunity to express my thanks to those who helped me with various aspects of conducting research and the writing of this dissertation. I would like to thank my wonderful committee chair, Prof. Diane T. Rover, for the great and fruitful discussions we had throughout my Ph.D. journey. It is a real pleasure working with Dr.Rover over the years. Additionally, I would like to thank my committee member, Dr. Matthew J. Darr, for providing me with the opportunity to solve a real world problem and propose a novel solution. I also would like to thank my friend and coworker, Dr. John Just for his help, advice, and thoughtful discussions with me. Additionally, I like to extend my thanks to my committee members. Last but not least, I would like to thank all friends, colleagues, the department faculty and staff for making my time at Iowa State University a wonderful experience.

ABSTRACT

Supervised learning is the workhorse for regression and classification tasks, but the standard approach presumes ground truth (label) for every measurement. In real world applications, limitations due to expense or general infeasibility due to the specific application are common. In the context of agriculture applications, yield monitoring is one such example where simple physics-based measurements such as volume or force-impact have been used to quantify mass flow. These measurements incur error due to sensor calibration. Using a vision system capturing images of a sugarcane elevator, mass of flowing material (bamboo and sugarcane) is accurately predicted from sparsely annotated images by training a deep neural network (DNN) in a semi-supervised fashion using only final load weights. The DNN succeeds in capturing the complex density physics of random stacking of slender rods as part of the mass prediction model, and surpasses older volumetric-based methods for mass prediction. Furthermore, by incorporating knowledge about the system physics through the DNN architecture and penalty terms, improvements in prediction accuracy and stability as well as faster learning are obtained. It is shown that the classic nonlinear regression optimization can be reformulated with an aggregation term with some independence assumptions to achieve this feat. Since the number of images for any given run is too large to fit on typical GPU vRAM, an implementation is shown that compensates for the limited memory but still achieve fast training times. The same approach presented herein could be applied to other applications like yield monitoring on grain combines or other harvesters using vision or other instrumentation and is by no means limited to the sugarcane application. Using a vision system with a relatively lightweight deep neural network we are able to demonstrate the generalizability of presented methods by estimating mass of bamboo with an average error of 4.5% and 5.9% for a select season of sugarcane.

CHAPTER 1. INTRODUCTION

This dissertation outlines the research done in pursuit of the PhD degree in Electrical and Computer Engineering at Iowa State University. This chapter presents motivating factors within the problem space and a concise statement of the problem.

1.1 Motivation and problem background

Advances over the past couple of years in machine learning algorithms, specifically deep learning algorithms and training methods, along with supporting open source frameworks such as Tensorflow has made leveraging complex high-dimensional datasets more accessible to scientists and engineers in various domains than ever before. Moreover, various methods in the domain of deep learning have been developed to accommodate the variability, structure, and complexity of different research problems.

Supervised learning [39] has been most widely used to solve various types of problems, but this method requires large amount of labeled training data to obtain decent accuracy levels. The process of labeling data can be expensive, difficult, and time consuming when dealing with very large datasets. In many real world scenarios it is common not to have ground truth measurement for every data point in the dataset, which poses a challenge to supervised learning.

Unsupervised learning [29] approach is intended to learn from data without any association with labels, leaving the algorithm to determine the data patterns completely on its own. This style of learning cannot directly achieve what supervised learning does, since there is no guarantee that the patterns the algorithm finds correlate directly with the ground truth or characteristics of interest; however, unsupervised learning is very useful in learning complex patterns, capturing statistical dependencies, and performing exploratory analysis as it can automatically identify structures in data.

While supervised learning is completely dependent on labeled data and unsupervised learning is completely independent, semi-supervised learning [10] sits in the middle between these two approaches. Semi-supervised learning employs learning techniques that are capable of performing a task like classification or regression with the presence of both, labeled and unlabeled data.

Mass flow estimation is of great importance to several industries and it can be quite challenging to obtain robust and accurate estimates depending on the material and application. Mass flow can be a critical parameter to industries like gas and oil, and it can be an optimizing factor to industries like mineral processing [78] and precision agriculture [54]. The mass flow estimation problem was investigated for several different applications with several different methods, (e.g.) works in [25, 2] attempted to measure mass flow using a machine vision approach. Further, other methods include using laser profilometry [74], ultrasonic sensors [22], radiation based sensors [79], and power sensors [31]. Yield monitors on harvesters are a key component of precision agriculture tools and mass flow estimation is the critical factor to measure. This allows for field productivity analysis, real-time adjustments to machine efficiency, and costs minimization of logistics by ensuring trucks are filled maximally without exceeding weight limits. Several common technologies used on grain harvesters, including impact plate sensors, are accurate enough on combines to be valuable but suffer from issues such as drift, and are not accurate enough for very dispersed material.

Sugarcane harvesting is a real life application where it is difficult to obtain a weight measurement for each sample. We are motivated by this agriculture application to devise, develop, and implement a deep learning approach that can accurately and robustly estimate the mass of flowing material (e.g. sugarcane). Further, we wish to prove the utility of the deep learning approach by showing that predicting the mass from images of a complex material flowing on a sugarcane elevator, is feasible and can replace older methods, while making a more accessible low-cost solution that only requires a regular camera and inference via a relatively light-weight deep neural network.

1.2 Thesis statement

My research work focuses on developing a deep semi-supervised learning algorithm to estimate mass flow of materials through a relatively simple 2D vision system offering a robust, accurate, and cost effective solution. This research problem can be formulated into three questions: (1) What are the deep learning techniques that can be utilized to capture a physical property (mass) of flowing material through only processing sparsely annotated images? (2) To what extend can transfer learning be applied from one application or material to another, and (3) How much of change is required and/or what requirements should be maintained to achieve generalization? To answer the stated questions, we follow a systematic procedure that goes into several phases of implementation. Figure 1.1 shows a sequential diagram of the project phases and each phase is discussed in details below.

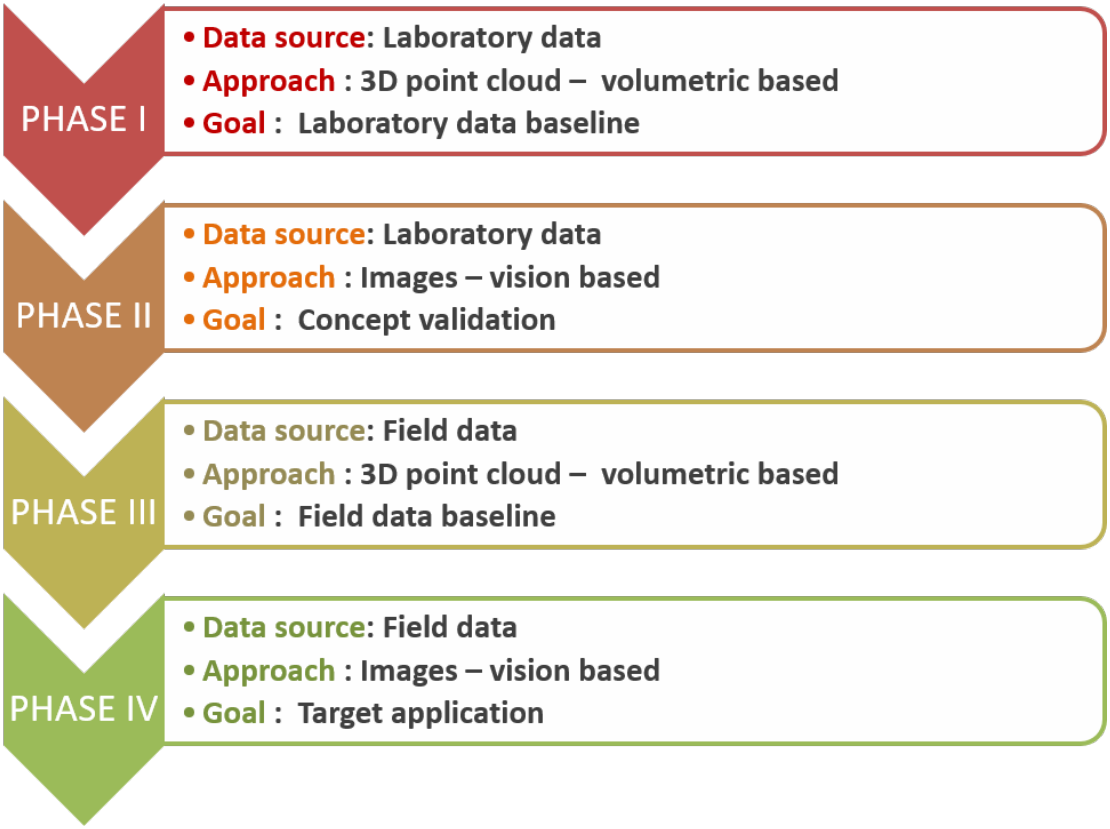


Figure 1.1: Summary of project phases

PHASE I In this phase, we evaluate a volumetric-based approach to estimate mass flow through generating a 3d point cloud volumetric representation of flowing material. In this approach, bamboo is used as a surrogate material to sugarcane under controlled environment setup and is maintained as a baseline comparison to a vision-based approach via deep learning.

PHASE II In this phase, we develop and implement a vision-based approach to estimate mass flow through a deep neural network that quantifies mass from images. This approach is conducted on bamboo and maintained under controlled environment setup to initially assess the feasibility of the vision-based method. Further, in this phase we compare against results obtained in phase I to support the development of a vision-based solution for sugarcane yield monitor on sugarcane harvesters.

PHASE III In this phase, a volumetric-based approach is evaluated to estimate mass flow through generating a 3d point cloud volumetric representation of flowing material. This approach is conducted on sugarcane under real environment operation and is maintained as a baseline comparison to a vision-based approach to estimate mass of sugarcane via deep learning.

PHASE IV This is the last phase of this research work and in this phase, transfer learning is applied to the task studied in phase II and then required changes are assessed accordingly in case there are any. This is a vision-based approach to estimate mass flow through a deep neural network that quantifies mass from images. This approach is conducted on sugarcane and maintained under real environment operation. Further, in this phase we compare against results obtained in phase III to examine if this method is superior as a decision towards the possibility of developing a sugarcane yield monitor in future work.

Figure 1.2 illustrates the dependencies between all the four phases. Boxes in orange represent intermediate results that are produced by certain phases and used in others. Green boxes represent final results which are the outcomes of this research.

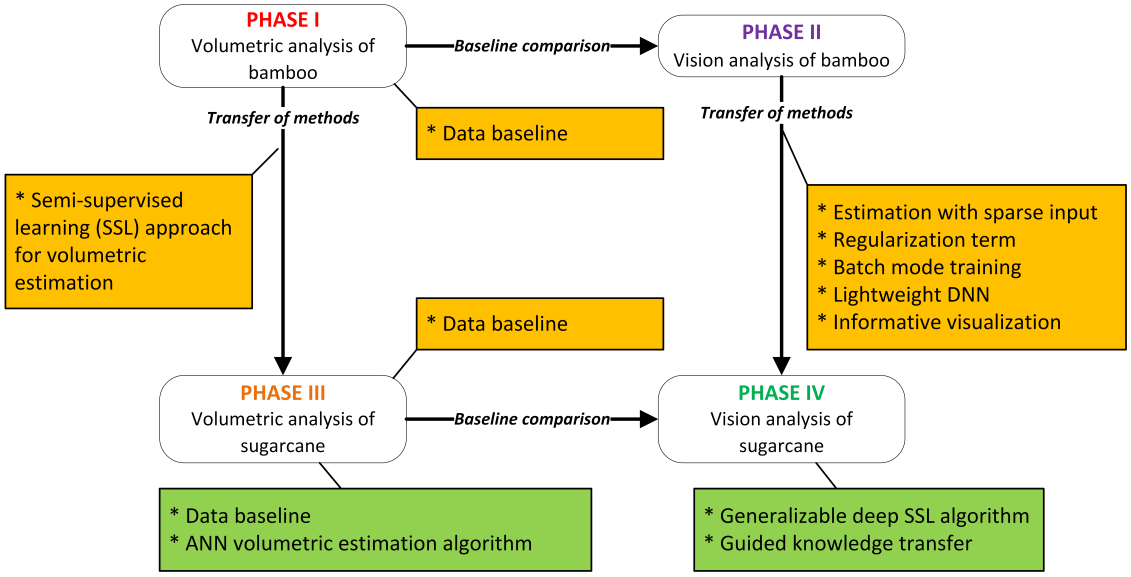


Figure 1.2: Phases dependency map

Figure 1.3 shows the proposed model of the SLEM (Semi-supervised estimation of mass) framework when deployed for real time mass estimation on an actual machine.

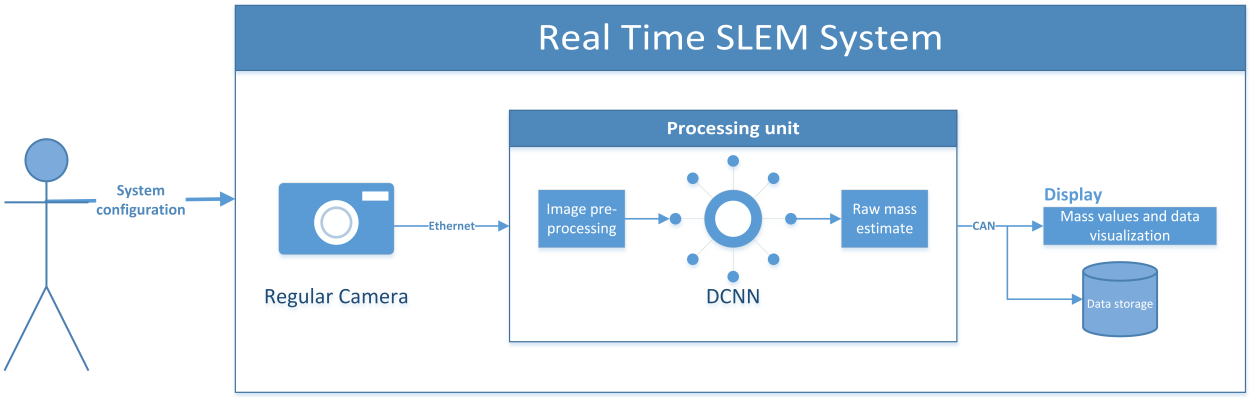


Figure 1.3: Mass estimation using deep learning (SLEM) framework

This work makes the following **contributions** as we progress from phase I to phase IV:

- Evaluating a volumetric method, baseline comparison, to estimate mass flow by estimating the underlying density of bamboo using statistical data transformation as well as artificial neural network working in a semi-supervised fashion. [**PHASE I**]

- Developing and implementing a deep learning vision-based approach to learn complex physics relationships between the bulk density, quantity, and location of material in images to accurately estimate mass, with only sparse ground truth that is tested on bamboo plant. [**PHASE II**]
- Devising and implementing a regularization term that enforces temporal smoothness to achieve more stable prediction signals and better accuracy on the data-point level. [**PHASE II**]
- Designing and implementing a training procedure that handles gradient computation in batch mode, since the number of images for any given run/video is too large to fit in vRAM on a typical GPU. [**PHASE II**]
- Designing and implementing a lightweight deep neural network (DNN) architecture that can run efficiently on low performance and memory bound devices.
- Evaluating an explanatory method of deep neural networks functioning when estimating mass under platform constraints. [**PHASE II**]
- Establishing a baseline that could be used for performance analysis in future studies. [**PHASE III**]
- Devising an algorithm that can serve as plug in tool in current industrial yield monitors that rely on a calibrated density to estimate mass flow. [**PHASE IV**]
- Techniques to generalize mass flow estimation methods to serve other tasks or materials. [**PHASE IV**]
- Evaluated rigorously on massive real life data (≈7000Mg) . [**PHASE IV**]

1.3 Organization of the dissertation

The remainder of this report is organized as follows. Chapter 2 describes the related work of developing yield monitors for sugarcane as well as the use of semi-supervised learning to solve sparse ground truth of vision-based problems. Chapter 3 shows the implementation of a volumetric-based approach to estimate mass flow of bamboo under controlled environment setup. Chapter 4 illustrates the implementation of a vision-based approach through a deep neural network to estimate mass flow of bamboo under a controlled environment setup. Chapter 5 shows the implementation of a volumetric-based approach to estimate mass flow of sugarcane under real environment setup. Chapter 6 describes the implementation of a transfer learning approach to adapt the vision-based approach described in chapter 4 to estimate mass flow of sugarcane under a real environment setup. Lastly, conclusion and future work are summarized in chapter 7.

CHAPTER 2. RELATED WORK

In this chapter we summarize the related work from two different perspectives. First, we discuss the related work from the application aspect where we reference several works that attempted to develop sugarcane yield monitor following variety of methods and we reference works that generally worked on solving the mass flow estimation problem. Thereafter, we discuss the related work based on the approach we are following to handle the sparsity problem. We reference couple of works that applied semi-supervised learning techniques to different applications and subsequently show the applicability of this method to our application. Further, we summarize some agriculture applications that utilized the power of deep learning to solve agriculture related problems.

2.1 Sugarcane yield monitor

Sugarcane is an important crop where it is an established source of sugar and one of the feed-stocks for efficient biofuel production [71]. Originally from Southeast Asia, sugarcane is now grown in tropical and subtropical countries around the world, including Brazil and a portion of the U.S. As a ratoon crop, it can be grown for 5 to 7 years and annually harvested without replanting [49], and even longer if not damaged during harvesting such that it continues to regrow. The crop grows year-round regenerating after cutting and is usually re-harvested at the same time period the following year. With this high throughput production, means of regular yield estimation are needed and that is where yield monitoring comes in. A yield monitor is a device consisting of a coupled set of sensors, which measure geographic and mass flow information during harvesting. This information can in-turn be translated to crop yield spatially within a field or over time. Thus far, yield monitors are also important in the process of filling trucks with crops by ensuring that trucks are filled with maximum allowed limit without overloading.

As human population continues to increase, pressure on the agricultural supply chain will increase. The same land must produce more food, while global competition pressures producers and suppliers to increase efficiencies in all areas of operations. To increase agriculture productivity while minimizing its environmental impact, the development of novel sensing technology and precision agriculture tools have arisen for generation and analysis of data to achieve the aforementioned objective. Adoption of current precision agriculture tools and technologies has led to higher yields, reduced environmental impact, reduced costs, and improvements in sugarcane quality [68] . Yield monitors have been a core component of precision agriculture and some of the earliest sugarcane yield monitors were developed in USA by Ag leader technology [1] and later was patented in 1994 [54] .

As an essential component of a yield monitor, a mass flow sensor provides feedback related to the amount of product at each point in the field where the harvester operates. This enables farmers to assess the performance in and between fields, and supports quantitative decisions such as when to replant or how to adjust inputs such as fertilizers. Additionally, mass flow sensors support real-time control of machine productivity by monitoring machine work against quantity harvested, or help in optimizing logistics such as filling semi-tractor trailers maximally without exceeding allowed limits.

There have been several techniques attempted over the last 20 years while trying to measure sugarcane yield on a machine during harvesting. Some examples include impact/deflection plate similar to what is ubiquitously used in grain crops, a weight plate in the elevator floor that triggers a reading as material flows over, optical sensors that measure the proportion of the elevator floor covered with material, and other methods that attempt to use measurements of the machine work performed and translate it into a mass flow. The overall strategy of methods can be summarized as shown in Table 2.1

Realizations of systems falling under these categories have been attempted with varying results. Authors in [16] used chopper and feed roller hydraulic pressure (indirect mass measurement) along with motor speed to calculate power exerted by the motors and volume of material through

Table 2.1: Summary of mass flow measurement techniques for harvesters

	Direct	**Indirect**
Mass Flow	Piezoelectric (Load cells, impact plate)	Calibration dependent on machine function(s)
Volume Flow	Displacement: Movable opening in contact with material. Area changes in proportion with amount of physical material flowing through it	Non-contact sensor that can measure physical dimensions of material

rollers. These measurements are calibrated with regression to predict mass flow. While simple and inexpensive since it requires very little change to the machine, wear on the machine and different crop conditions may all shift the response.

Earl et al. [20] developed a yield monitor system that rely on a weight measurement to assist sugar beet harvesters. The downside of this method is the need of several instrumented trailers to be located with each harvester to minimize the harvesting operation cost. Authors in [59] followed a similar approach to work in [20], where they constructed a mass measurement trailer with a load cell equipped with differential global position system (DGPS) and a data acquisition system (DAQ). The system produced reasonable results but still had the same limitations as in [20].

Mailander et al. [50] designed a basic monitoring system that did not contain an accelerometer to offset the mass and momentum changes of the weight plate as the machine moved through the field or tilt sensors to correct the calibration equation. The system resulted in an average percent error of 11%. Authors in [15, 56, 14, 53, 7, 48] tried a version of a weight plate with load cells in the floor of the elevator. Adjustments for pitch of the elevator were made with an accelerometer, and readings triggered as material flowed over the plate. It is a direct mass measurement but it requires significant changes to the machine, is costly and complex, and is susceptible to mechanical noise. Additionally, load cell drift due the moving parts of the harvester and can suffer from buildup of material jamming the plate.

Other methods that have been investigated to estimate sugarcane yield include force impact. Work in [80] described a yield monitor system that uses force to estimate yield of cane through

utilizing a deflection plate sensor and a control monitor. The deflection plate is placed near to the top end of the elevator after the billets exit to fall into the wagon. The measured force and elevator speed are calibrated to predict mass flow. Another approach to estimate yield by measuring the stem-bending force was tested by Mathanker et al [51]. Load cells were placed between two parallel pipes making a push bar that was installed between crop dividers. The system showed an R^2 of 92% between forces on the push bar and yield when harvesting napiergrass, but the system could not withstand the large impact forces encountered during the harvest of sugarcane.

An under-elevator optical sensor array was tested by Price et al. [61]. They measure the amount of time the sensor array was covered with material and then calibrate the calculated duty-cycle to mass flow. This is an indirect volumetric method which is relatively simple and inexpensive. However, it depends on ambient light and a reliable stacking of material since it attempts to quantify volume using only two dimensions. Additionally, the calibration is highly affected by changes in material density. In further studies, Price et al [62] developed a fiber optic yield monitoring system that measures the volume of sugarcane to estimate yield. Under dry field conditions the system resulted in an average error of 7.5%; however, the system performance degraded under wet and muddy conditions, where about 75% of sensor readings were lost. Moreover, Price et al. [60] also developed an alternative optical yield monitor system that uses two laser distance sensors mounted above the loading elevator to measure height and length of the billet piles per slat. Different methods were attempted to find the best relationship between volume and material weight. It was found that the cumulative billet pile length had the best relationship to harvested weight. An R^2 ranging from 93% to 97% was reported. The system is relatively simple and easy to install; however, the system still suffered from the piling up of debris and sugarcane leaves.

To our knowledge, there has not been research that used an indirect volumetric based approach with point clouds to estimate the yield of sugarcane like what is discussed in this work; however, Jadhav et al. [33] developed a volumetric mass flow sensor that uses a LiDAR sensor to estimate

the total mass of citrus. Their system was tested on an inclined and a horizontal conveyor like those used on mechanical harvester and debris removal systems. Results show that the system can estimate mass flow with an average error of 7% and a standard deviation of 7% for the incline conveyor, and an average error of 7% and a standard deviation of 5% for the horizontal conveyor. Several direct and indirect volume-based measurement systems were also explored by Schmittmann [35] in a small trial for sugar beet and potato yield. They reported good results with a laser scanner ($< 4\%$ error) and mechanical fingers, but no further information could be found of a product developed from their work.

2.2 Machine and deep learning and computer vision

Machine learning and computer vision systems have been heavily applied in many fields [72]. Many agricultural applications employed these approaches for purposes such as size estimation of citrus fruit [67], crop yield prediction and nitrogen status estimation [12], obstacles and anomalies detection [13], and root-soil segmentation [19]. The use of machine learning in vision systems has shown much better performance than traditional computer vision approaches [41]. Machine learning is the workflow in which features are manually or automatically selected and therefore a model is trained. A subset of machine learning described as deep learning offers automatic learning of features. Indeed, deep learning [40, 23] is specifically focused on the use of artificial neural network (architectures that comprise multiple layers) and its variants.

With complex deep learning models, complex problems can be solved efficiently. Image processing is an important research area in the domain of agriculture and intelligent data analysis techniques such as image classification/identification are used in various agricultural applications [76, 66, 69]. Some of the applications that deep learning models were employed in include plant classification and identification with convolutional neural networks [82, 42], plant disease recognition by leaf classification [70], classification of land cover and crop types [37], and crop yield prediction with convolutional neural networks [55].

There are multiple types of learning in deep or machine learning such as supervised learning (the most common learning approach) [45], semi-supervise learning [10], unsupervised learning [5], and reinforcement [81] learning. In this work we focus on the semi-supervised learning approach. Semi-supervised learning is a learning paradigm concerned with learning in the presence of labeled and unlabeled data. Semi-supervised learning has been extensively investigated in the last decade [88]. This type of learning is motivated by the need for an alternative to the expensive, time-consuming, and tedious process of labeling data as well as to make up for missing ground truth signals from real world applications. Semi-supervised learning has gained a lot of interest and several works have been done to drive this approach of learning to solve problems through either proposing general learning models or application specific techniques.

This type of learning was employed in applications such as crowds counting [73, 9], network traffic classification [21], person identification in webcam images [4], weed mapping in sunflower crops [58], monocular depth map prediction [38], text chunking [3], and object detectors from videos [52]. Some of these applications are a classification task and some are a regression task. Indeed, semi-supervised learning is less studied for regression tasks compared to classification ones and the reason is that in a classification task the number of categories is finite, whereas in regression the object values are continuous. Authors in [86, 83] introduced domain knowledge to formulate a regularization term, which utilizes the underlying structure of unlabeled data for regression and dimension reduction. Work in [85] introduced a learning style described as co-training where two different regressors make their estimations on unlabeled data and refine the performance by learning the results of each other.

Some of the popular semi-supervised learning models [87] are self-training [63], mixture models [17], co-training [85], multi-view learning [44], and semi-supervised support vector machines [6]. The common theme between these learning techniques is that they focus on the input data in the process of correlation with the expected response. In other words, the input data is modeled in such a way that certain information (underlying structure of the data, data clusters, or data regularization and normalization) is leveraged to guide in the process of prediction. In our

proposed method we focus on the response itself and drive the feature extraction process. We utilize the underlying relationship in the response signal, where the unlabeled data relates to the labeled data in the sense is that over a period of time, a single label represents the ground truth sum of all missing ground truths of each single output; alternatively, to take advantage of measurements with sparse ground truth to train a predictive model, a tractable relationship must be asserted and hold between the measurements and labels. As long as that relationship can be formed into a loss function that can be optimized, a model then can be trained.

CHAPTER 3. VOLUMETRIC-BASED MASS FLOW ESTIMATION UNDER CONTROLLED ENVIRONMENT OPERATION

In this chapter we describe a volumetric-based approach to estimate mass flow of flowing material in a controlled environment setting (laboratory setup). To our knowledge, there has not been research that used an indirect volumetric-based approach with point clouds to estimate the yield of sugarcane; however, Jadhav et al. [33] developed a volumetric mass flow sensor that uses a LiDAR sensor to estimate the total mass of citrus. Their system was tested on an inclined and a horizontal conveyors like those used on mechanical harvester and debris removal system. Results show that the system can estimate mass flow average error and standard deviation of error of 7% and 7% respectively for the incline conveyor, and 7% and 5% respectively for the horizontal conveyor. Several direct and indirect volume-based measurement systems were also explored by Schmittmann [35] in a small trial for sugar beet and potato yield and reported good results with a laser scanner ($< 4\%$ error) and mechanical fingers, but no further information could be found of a product developed from the work.

We study and extensively explore a volumetric approach to estimate sugarcane yield by generating point cloud of the material on the elevator using a stereo camera. Stereo cameras create a full 3d point cloud of the material, unlike laser scanners which capture a 2d plane with each measurement and miss material in between each measurement. Thus far, the intended goal of the work presented in this chapter is to create a baseline comparison to our proposed vision-based method, which is presented in chapter 4.

3.1 Materials and methods

3.1.1 Fundamentals of operation

The basic operation of the yield monitor discussed herein consists of measuring the volume of flowing material on the machine elevator, and converting the measured volume to a mass via a calibrated density. Thus it is an indirect volumetric method per Table 2.1. This is done with a stereo camera instead of LiDAR for cost effectiveness, robustness to dust and dirt, and a much larger measurement area which does not miss any material.

At a typical elevator speed of 2 m s^{-1}, a LiDAR line would measure an $x - z$ cross section of the elevator (in the same plane as the slats) to obtain depth z measurements at equally spaced horizontal x distances across the width of the elevator. An average sensor sampling at 25Hz will only acquire a cross-sectional measurement of the material spacing of 8cm ($200\frac{cm}{s} \times \frac{1}{25}s$) along the direction that the material is conveyed on the elevator. Thus it would be necessary to interpolate in the y direction between $x - z$ cross sectional measurements to create a surface from which to calculate volume. However, a more sophisticated LiDAR would be able to cover more area, but that comes with much more costs compared to a stereo camera. Further, the harvesting environment is harsh and full of dust and LiDARs are generally sensitive to dust. A stereo camera, on the other hand, is solid proof against dust and capable of producing a dense 3d point cloud over a large surface with better resolution of the surface of the material. In this case the system operated at 7.5Hz, which was enough for overlapping successive point clouds and ensuring no information loss.

The mass flow, up to a scaling constant depending on the time period to estimate over, is shown in Equation 3.1. The stereo camera estimates the volume within the region of interest (ROI), which is denoted as V_c and has units of meters cubed per meter up to a scaling constant. Then this quantity is scaled by the distance the elevator moves ($\Delta t \times v_e$) in between the next volume estimate to find an incremental accumulated volume. A simplifying assumption inherent in this formulation is that the volume calculated (V_c) is spread evenly across the ROI, since the

incremental accumulated volume V_Δ is directly proportional with the distance the elevator moves. The incremental volume is then converted to mass via a multiplier (calibration factor/density) as seen in Equation 3.1. Any error in density of the material can be seen to directly contribute to error in predictions of mass, and therefore yield as shown in Equation 3.2

$$V_\Delta = \Delta t \times V_e \times V_c; \quad m_\Delta = V_\Delta \times \rho \tag{3.1}$$

where:

V_Δ = incremental accumulated volume (m^3)
V_c = proportional to cross sectional area of material in current frame from stereo camera (m^2)
Δt = image capture time (s)
V_e = harvester's elevator velocity level ($m\ s^{-1}$)
m_Δ = incremental accumulated mass (kg)
ρ = density ($kg\ m^{-3}$)

$$Yield = \frac{\dot{m}}{w \times v_m} \tag{3.2}$$

where:

$Yield$ = crop yield ($Mg\ ha^{-1}$)
$\dot{m}$ = mass flow ($kg\ s^{-1}$)
V_m = vehicle speed ($km\ h^{-1}$)
w = row width (m)

3.1.2 Laboratory setup

Extensive proof of concept testing was conducted prior to field exposure (phase III of this work - chapter 5). Experiments were designed to test the system as closely as possible to factors present during typical operation, while controlling for factors outside of system control such as particle density. To accomplish this, bamboo was used as a surrogate material to sugarcane since it has stable material properties long term (will not rot or dry out) while being similar in shape to

sugarcane. Bamboo was conveyed into a sugarcane elevator at various flow rates with a stereo camera mounted on the elevator. Figure 3.1 shows the overall laboratory system setup.

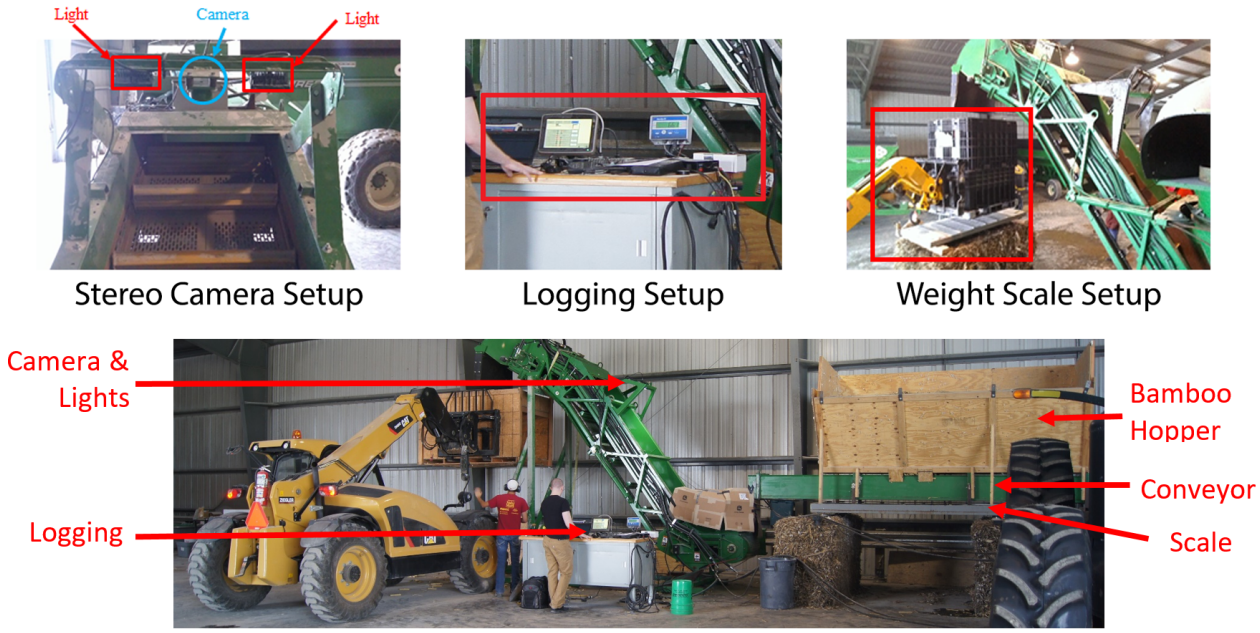


Figure 3.1: A picture of the overall system of laboratory setup for volume measurement testing

The system includes hydraulically driven elevator and conveyor, logging system, stereo camera system (imagers with 120dB HDR and a global shutter – discontinued), and ground truth weighing scale. Initially, a weight measurement (ground truth) is taken using the scale, then the material is conveyed via the conveyor and elevator. As the material travels, the stereo camera produces a 3d point cloud of the material that is converted to volume via a matching algorithm. Lastly, the logging system records the volumetric data and stores it on a storage unit.

Since lighting is the main signal for a camera, non-optimal lighting can cause poor signal to noise ratio (too high, or too low). Further, Motion and sub-optimal lighting can cause blurring. Our goal in the lab experimentation was to simulate a wide range of these factors including elevator speeds and mass flows (since objects further/closer to the camera also affect apparent motion) for many short runs in which the same weight/volume of material is run through the system and gauge repeatability and reproducibility (R&R). Ideally we calculate the same accumulated volume

across all range of variables thus showing the system is robust against these factors. The end goal was to observe the same amount of volume measured from the camera in each test.

Six lighting levels were used that span from $\approx$ 0.7k lux up to $\approx$ 6.7k lux as measured by a light meter located on the elevator in front of the camera. The elevator speed was measured by an inductive proximity sensor on the drive sprocket and varied between 1.0 m s^{-1} to 2.2 m s^{-1}. Runs length spanned between 20 seconds up to 120 seconds and the total mass per run ranged between 230kg up to 300 kg. These factors are also summarized by histograms as shown in Figure 3.2.

A total of 239 runs were obtained, which include 8 empty runs (zero mass) running for $\approx$ 60 seconds with the elevator moving to be sure the system was unbiased. Elevator speed was measured by an inductive proximity sensor on the drive sprocket. A run in this context represents a sequence/stream of volume measurements (produced by the stereo camera matching algorithm) captured for a weighted amount of material. A run is also referred to as a wagon load.

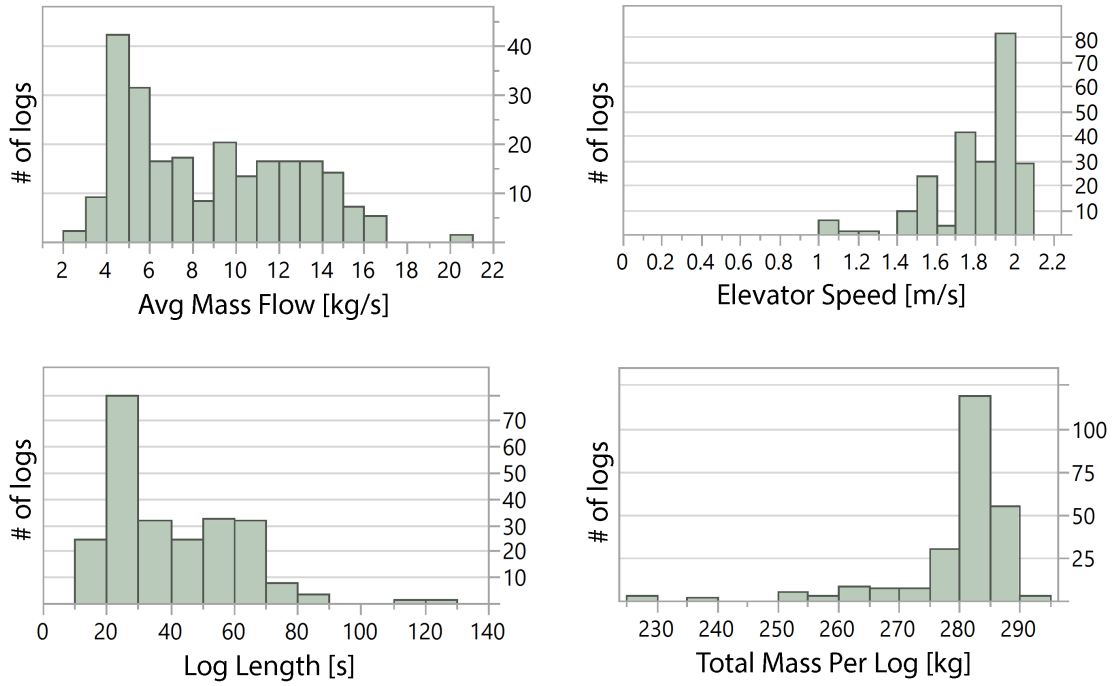


Figure 3.2: Summary of data from testing runs showing the range of factors covered - not including empty runs

3.1.3 Mass estimation methods

Two approaches were devised to estimate mass, where the first approach aims to applying a simple monotonic transformation to the volume measurements to reduce dispersion and then estimate the underlying density [calibration factor], where density is calculated as the total ground truth mass of the material divided by the accumulated transformed volume (sum of V_Δ from Equation 3.1). The density values are then used to produce instantaneous mass measurements per frame as shown in Equation 3.3).

$$m_\Delta = \rho \times xfm(V_c) \times V_e \times \Delta t \tag{3.3}$$

In this formulation ρ is the underlying estimated density and "xfm" is a monotonic transformation that is applied to volume (V_c), scaled by elevator speed (V_e) and time between measurements (Δt).

The other approach implies estimating mass directly where density is implicitly predicted using a feed-forward neural network. The algorithm explicitly uses the incremental volume to predict mass, and then scales it by elevator speed and capture time as shown in Equation 3.4.

$$m_\Delta = f(max(V_c - \beta, 0); \theta) \times V_e \times \Delta t \tag{3.4}$$

In this formulation, f is a 2-layer neural network parameterized by θ that outputs a prediction of density based on the point cloud volume (V_c), scaled by elevator speed (V_e) and time between measurements (Δt). The β-parameter is also fit to account for any positive bias present since the stereo volume calculation was designed towards not missing any volume. This was also a meaningful formulation since if the β parameter turned out to be negative then it would indicate a volume estimation in need of refinement, since it would be compensating for volume not detected.

3.2 Results and analysis

As per the design of experiments, lighting was one factor that was controlled during testing. It was found that so long as lighting was kept at a certain minimum level (in this case $> 2.7k$ lux),

significant nonlinearities with the point cloud estimation were avoided. In low lighting, the stereo camera matching algorithm was found to produce a higher number of points for the point cloud overall, but the points tended to have a lower-than-expected elevation as measured from the plane of the elevator. Since the shutter speed had to be fast due to the quick movement of the slats (and therefore material), increasing the exposure time of the camera was not an option.

Figure 3.3 shows the average volume of material per frame against density. The data points circled in red were collected under light levels below 2.7k lux, hence a significant decreasing density trend is observed as the volume increases. Thus far, as long as certain minimum amount of lighting is provided, a decreasing density still persists but is not as extreme as shown by the data points circled in green in Figure 3.4. This remaining decreasing density trend was hypothesized due to a packing effect where the measured bulk density of the material decreases as the total volume of material increases. This is supported by the following works [46, 84], which investigated the densities of disordered packing of slender cylinders.

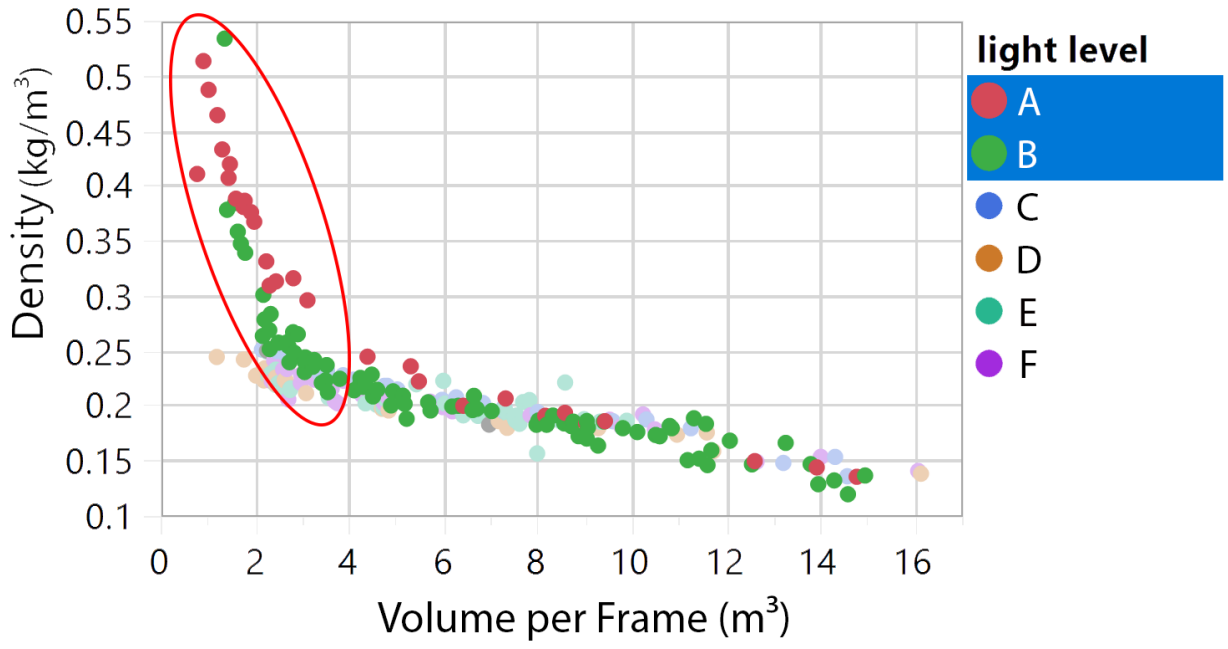


Figure 3.3: Density dependence trends. Light levels (A & B): poor sensitivity of stereo camera at lower volume flows with light level less than 2.7 lux

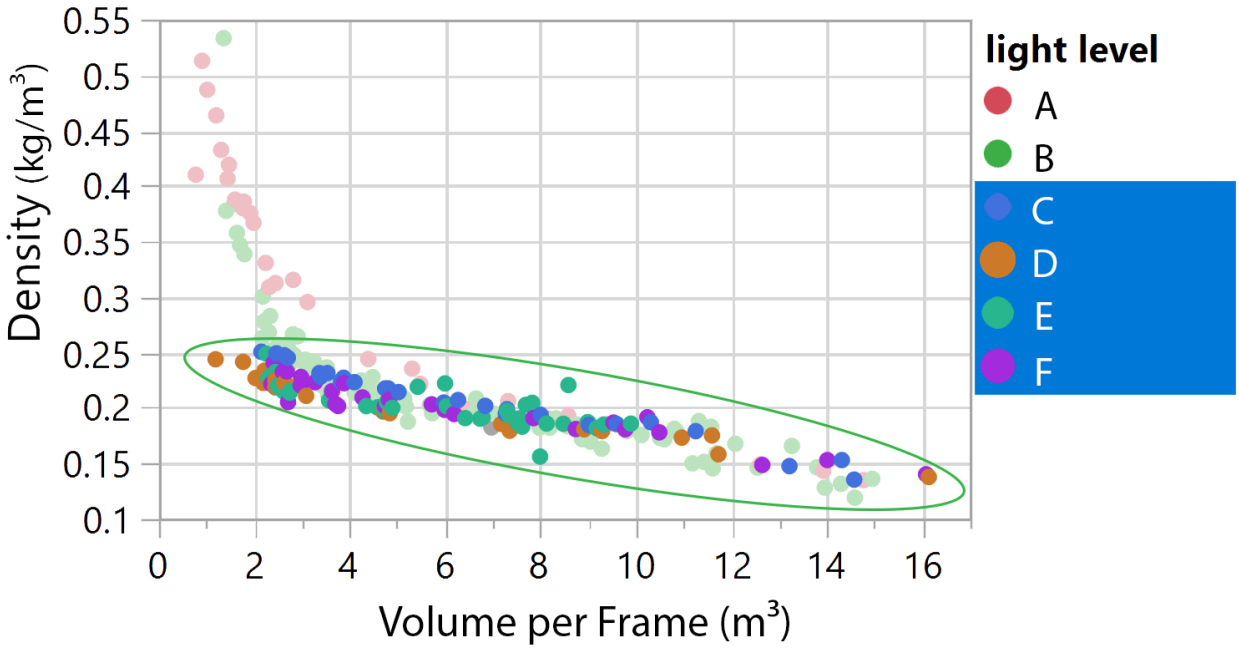


Figure 3.4: Light levels (C, D, E, & F): runs with adequate lighting show decreasing bulk density with volume likely due to disordered stacking of material – a real/physical phenomenon

Note that the variation increases for volume flows larger than $12m^3$ per frame since at that level the bamboo begins to overflow the slats and fall back down the elevator, resulting in double counting and lower fidelity point cloud estimation. This can also be seen in Figure 3.6. Bamboo is also especially slick when compared with sugarcane, which is less likely to experience this issue due to higher friction and generally not often reaching such high flow rates.

Since the monotonic transformation approach relies on estimating the mass based on estimating the density, we consider evaluating the coefficient of variation (CV) of density values, since density is the calibration factor that is used to predict mass. The CV of the density values was 12% and transforming the volume shaved $\approx 2\%$ off of the CV of the density values, bringing it from 12% to 10%, with the histogram comparison shown in Figure 3.5. It is important to minimize the coefficient of variation of density values since the lower the value of the coefficient of variation, the more precise the estimate.

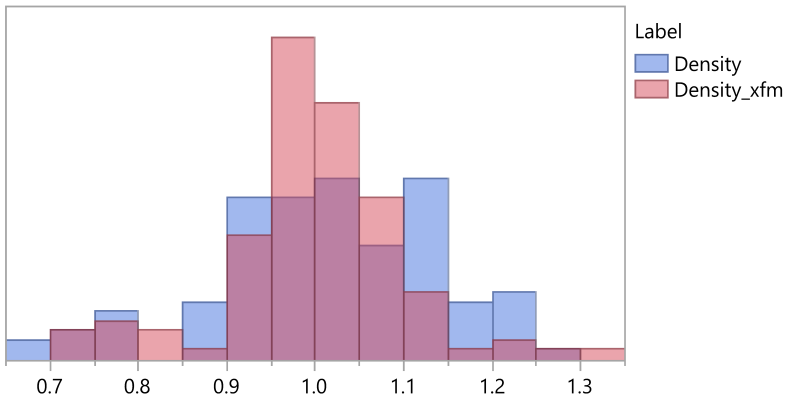


Figure 3.5: The overlaid density histograms show the transformed volume concentrates the densities more and reduces the CV from 12% to 10%

As the primary task of a yield monitor is to predict mass flow, the controlled lab testing provided a means to assess the sensor accuracy in this regard without introducing variation from external factors like material density changes. The average predicted mass flow (utilizing a monotonic transformation on the volume to estimate density) was plotted against the average actual mass flow for each run to observe correlation in Figure 3.6.

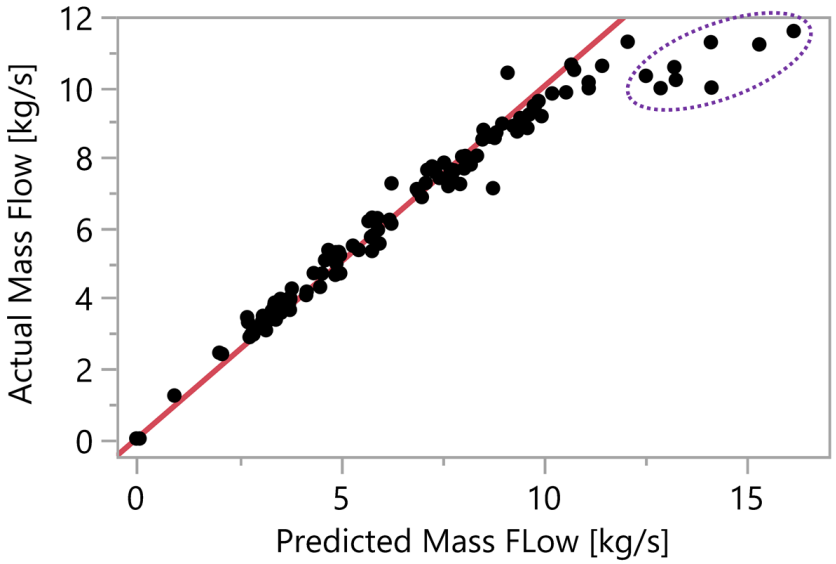


Figure 3.6: Predicted vs. actual mass flow shows a strong trend indicating the potential of this method as a yield monitor. The circled points represent flow levels beyond system capacity, where the bamboo begins to overflow the slats and slide back down the elevator

With the intercept forced through zero, the linear correlation can be seen to be very strong, achieving an R^2 of 97.4% when runs with overflowing material are ignored. With these results showing strong potential, the system was considered good enough to pass to field testing.

With the neural network, we were able to directly estimate mass and only need to scale it with elevator speed and time between measurements to get an instantaneous mass measurement. We evaluate the neural network performance using mean square error (MSE). Lab testing with neural network estimation achieved an average error of 8.65% while it achieved an average error of 14.65% when evaluated using monotonic transformation. We notice the advantage of using a shallow neural network over a simple monotonic transformation.

We further investigate the system performance under different lighting conditions using neural network. Overall, the system achieved an average error of -1.5% with a standard deviation of 18.1% for all runs including those identified to have too low light levels to have volume estimates on. Without including the low light runs the system achieves an average error of -4.8% with a standard deviation of 11%. The large variance implies the sensitivity of the volumetric system to variable lighting conditions. Figure 3.7 shows an overlay of histogram distributions of error, where red histogram does not include low light runs and blue histogram includes all runs from laboratory data.

Results summary

- ✓ Established a baseline comparison using 3d point cloud of volumetric data for laboratory data (bamboo).
- ✓ The system achieved a CV of 10% when using monotonic transformation.
- ✓ Ignoring the runs with material overflow, the system achieved very strong coefficient of determination R^2 of 97.4%.
- ✓ Neglecting runs with low light (<2.7 lux) the system achieved an average error of -4.8% with a standard deviation of 11% .

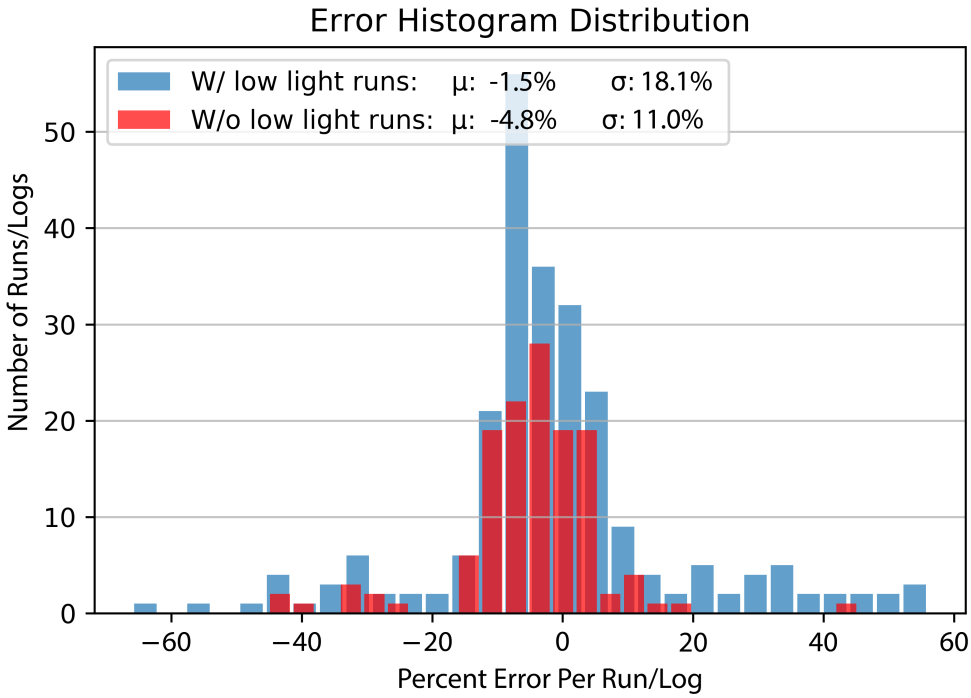


Figure 3.7: Overlaid error histograms of laboratory data using volumetric estimation showing smaller standard deviation of error when excluding runs captured under low light (<2.7 lux) setting

3.3 Conclusion

In this chapter we evaluated and discussed a volumetric based approach to estimate mass of flowing material (bamboo). Mass is inferred by estimating the underlying density [calibration factor] of material when applying monotonic transformation to the volumetric data. Using a neural network offered direct estimation of mass thus more precise estimate. The system achieved an R^2 of 97.4% when fitting average volume flow per test against average mass flow after correcting for bulk density changes with volume. Results reported in this system [PHASE I] will be used as a baseline comparison to a vision-based method under controlled environment setup [PHASE II - Chapter 4].

CHAPTER 4. VISION-BASED MASS FLOW ESTIMATION UNDER CONTROLLED ENVIRONMENT OPERATION

In this chapter we discuss the possibility of estimating mass via an end-to-end deep neural network framework from sequences of images. A thorough background about deep learning and how it is leveraged to solve our problem is given. We utilize the bamboo dataset under controlled testing (presented in PHASE I - Chapter 3) to examine the viability of our proposed vision-based method. We essentially work with the bamboo dataset for several reasons; First, to assess the viability of the proposed method, it is preferred to work with a problem with less complex space. Second, we would like to profoundly understand how the neural network operates on a less complex problem in order to transfer the methods to a more complicated problem as well as generalize to other similar problems. Implementation of this work can be found on the following GitHub repository MSE [26].

4.1 Deep learning background

This section gives a brief background about deep learning and is intended for those with little to no background about deep learning. If you have background on deep learning, then you can pass onto the next section without having to read through this section.

The term AI or artificial intelligence has been around for decades and it is widely known even by out-of-field individuals. Machine learning is an area of AI that is focused on providing systems with the ability to learn and improve automatically without explicit programming and only by relying on experience. Alternatively, machine learning focuses on creating algorithms which can modify themselves without human intervention to produce the desired output by utilizing data to teach themselves.

Deep learning (DL) is a subset of machine learning with algorithms that function in a similar way to those in machine learning, but there are multiple layers of these algorithms. Each layer of deep learning algorithms provide different interpenetration of the data it feeds on. Deep learning is more specific to artificial neural networks [89] where there are multiple hidden layers, and that is why deep learning is described as deep.

As a branch of machine learning, deep learning employs the layers of artificial neural networks to process data, visually recognize objects, and understand human speech. Information flows through each layer with the output of the previous layer providing input for the next layer. The first layer in the artificial neural network is called input layer and the last layer is called output layer. All layers that fall in-between the input and output layers are referred to as hidden layers. Each layer is typically a simple uniform algorithm containing a specific kind of non-linearity function.

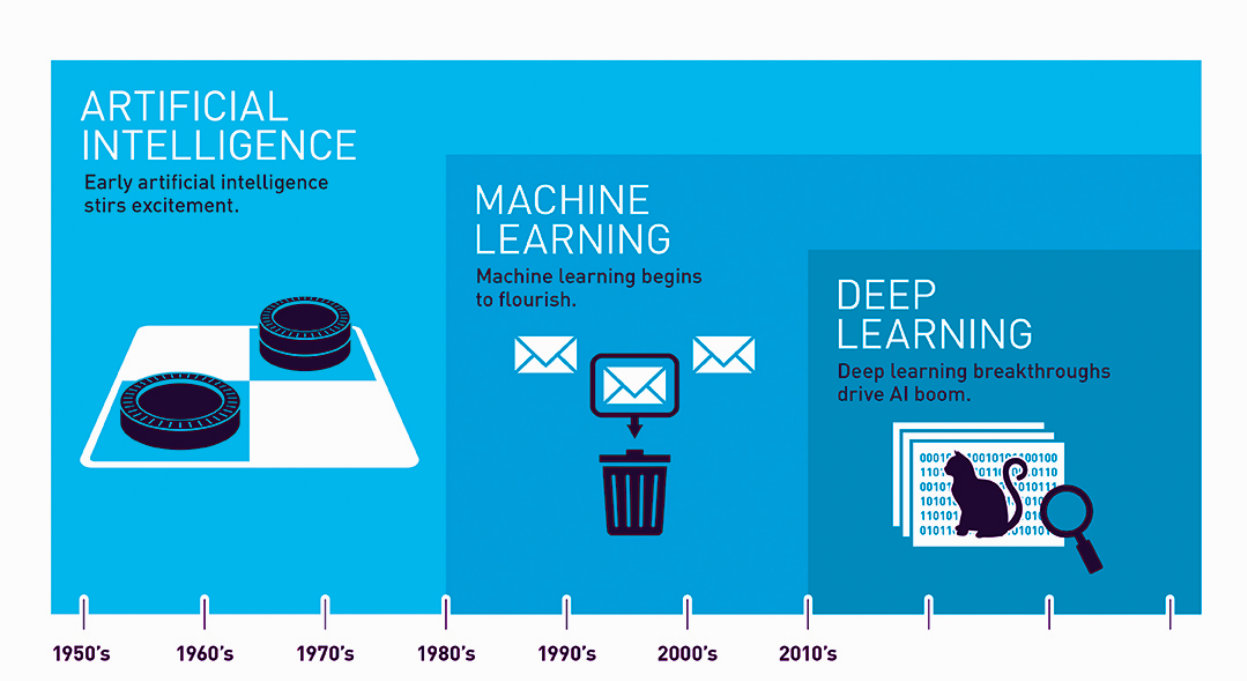


Figure 4.1: Artificial intelligence vs. machine learning vs. deep learning - adapted from [47]

4.1.1 Learning paradigms

Deep learning algorithms can learn from data under different learning paradigms. These learning types are mainly identified as supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

Supervised learning is a learning paradigm where inputs and corresponding labels are provided. Inputs are processed by the network and the resulting outputs are compared against desired outputs/labels using a loss function. Errors induced from the loss function are then propagated back through the network causing the network to adjust parameters (known as weights and biases) that control the network. This process occurs over and over as the network parameters are continually tweaked to minimize the error. Supervised learning solves primarily two types of problems, classification (finite values/classes) and regression (continuous values).

Unsupervised learning is a learning paradigm where inputs are provided without any corresponding labels. The algorithm is responsible for determining the data patterns completely on its own. This is often referred to as adaptation. This style of learning cannot directly achieve what supervised learning does, since there is no guarantee that the patterns the algorithm finds correlate directly with the ground truth or characteristics of interest; however, unsupervised learning is very useful in learning complex patterns, capturing statistical dependencies, and performing exploratory analysis as it can automatically identify structures in data. Unsupervised learning is usually used for clustering purposes.

Reinforcement learning is a learning paradigm that learns from mistakes. The algorithm in reinforcement learning works in an environment where an agent takes decisions to maximize the cumulative reward and improve the learning efficiency. The difference between reinforcement learning and supervised learning is that in supervised learning the training data has the answer key with it hence the model is trained with the correct answer itself. On the other hand, in reinforcement learning, there is no answer in the training data but the reinforcement agent

decides what to do to perform the given task. In other words, in the absence of training dataset, the agent is bound to learn from its experience.

Semi-supervised learning is a learning paradigm that involves learning from labeled and unlabeled data simultaneously. While supervised learning is completely dependent on labeled data and unsupervised learning is completely independent, semi-supervised learning sits in the middle between these two approaches. This approach is motivated by the fact that labeled data is often costly to generate, whereas unlabeled data is generally not. In the training process, the number of unlabeled samples is often larger than the number of labeled ones. Semi-supervised learning employs learning techniques that are capable of performing a task like classification or regression with the presence of both labeled and unlabeled data.

4.1.2 Deep learning algorithms

Deep learning is generally wrapped around neural networks with multiple hidden layers. There are several variants to deep neural networks that are designed to tackle certain tasks. We summarize some of these variants as follows.

Artificial neural network (ANN) is the simplest form of neural networks. They are referred to as feed-forward or multi-perceptron networks. ANN models are inspired by biological neural networks based on the functionality of neurons. This type of neural networks is often suited for tabular data. Figure 4.2 shows an example of a 4-layer ANN.

Deep belief neural network (DBN) is another variant of neural networks and is a generative graphical model composed of multiple layers of latent variables. Connections in DBNs exist between layers but not between units within each layer. Deep belief networks are typically used to recognize, cluster, and generate images, video sequences and motion-capture data. Figure 4.3 shows an example of a 7-layer DBN.

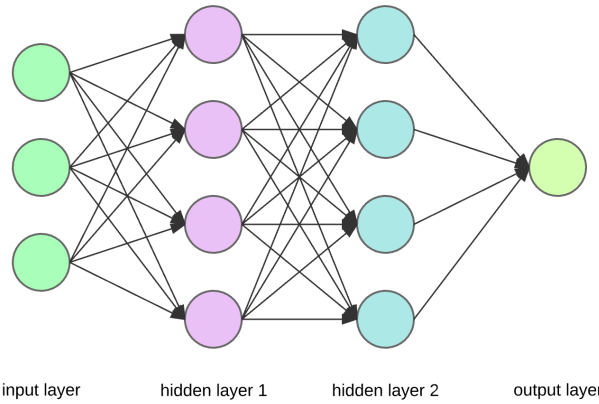


Figure 4.2: Example of a feed-forward artificial neural network - adapted from [43]

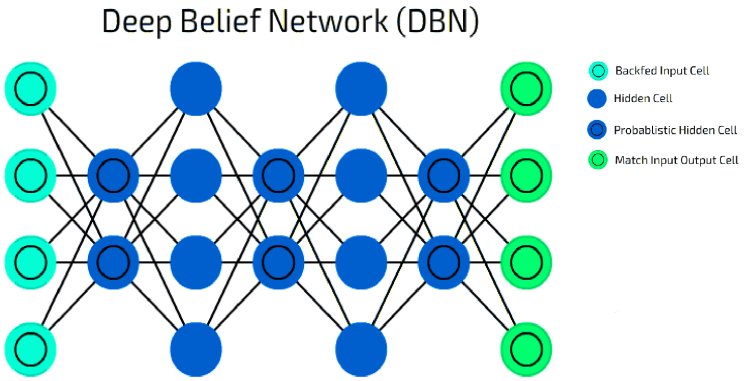


Figure 4.3: Example of a deep-belief neural network - adapted from [8]

Recurrent neural network (RNN) and LSTMs is a class of deep neural networks that is used for sequential data processing, text mining, speech recognition and natural language processing in general. In recurrent neural network, connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Unlike feed-forward neural networks, RNNs can use their internal state to process sequences of inputs. LSTM , stands for long short term memory, is a special kind of RNNs and is a better version of recurrent neural networks that solves a problem identified as vanishing gradients [30], which RNNs suffer from. Unlike RNNs, LSTMs are capable of learning long-term dependencies. Figure 4.4 and Figure 4.5 show an example of an RNN and an LSTM respectively.

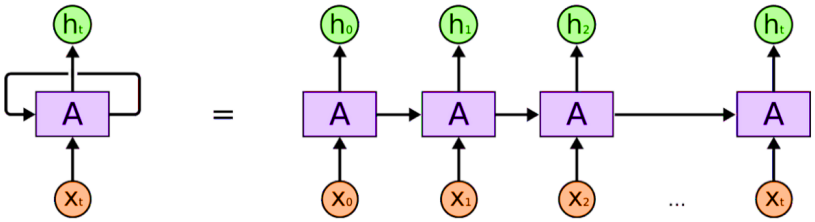


Figure 4.4: Example of a recurrent neural network - adapted from [36]

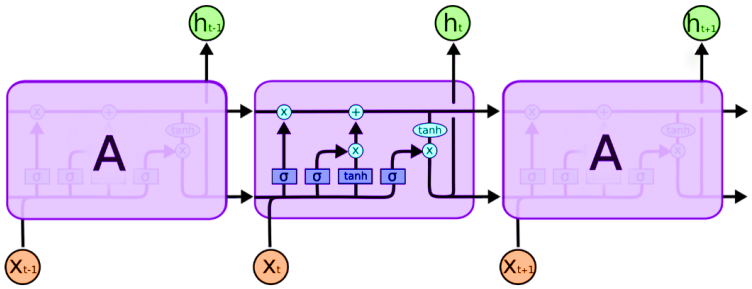


Figure 4.5: Example of a long term short memory network - adapted from [36]

Convolutional neural network (CNN) is a very common type of neural networks that is specialized in image processing. CNNs get their name from a mathematical operation called convolution and that is due the nature of how they process image data. CNNs are excellent feature extractors, where these features are eventually used to serve a classification or a regression problem. In our work we use this type of neural networks to solve our vision-based problem. Figure 4.6 shows an example of a CNN.

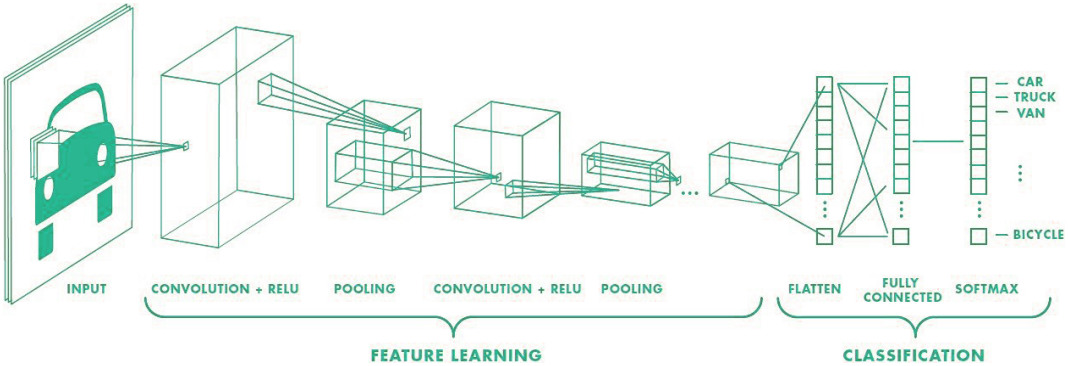


Figure 4.6: Example of a convolutional neural network - adapted from [65]

4.1.3 Convolutional neural networks overview

In this subsection we describe the building blocks of convolutional neural network as well as illustrate the functional components of this algorithm. The primary components of typical convolutional neural network are convolutional layers, pooling layers, and fully connected layers. A brief explanation of these layers is shown as follows.

Convolution layer is the main building block of a convolutional neural network. This layer is composed of several filters/kernels that are used to extract certain information, e.g. lines, edges, corners, etc, and store it in what is called feature maps. Extracted information is used later in the prediction process. The convolutional layer essentially performs a mathematical operation called convolution that involves 3-dimensional multiply accumulate (MAC) operations across the columns and rows of input images. Figure 4.7 shows an example of a convoluted output using a filter that is applied to the input images. Equation 4.1 shows how the convolution operation is done to calculate the outputs in the activation map/convoluted output.

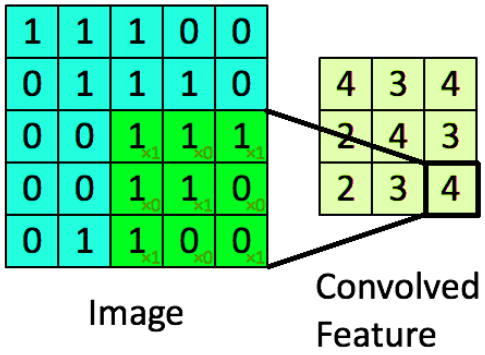


Figure 4.7: Example of convolution between 3×3 filter and 5×5 input image - adapted from [65]

$$OUT^{i}_{neuron} = \sum_{j=1}^{k} INPUT^{ij} \times weight^{ij} + Bias^{i} \tag{4.1}$$

Pooling layer is basically a form of nonlinear sub-sampling that is used to reduce the feature dimensions as we go deeper in the convolutional neural network. Alternatively, pooling layer is used to control the size of extracted information from convolutional layers as well as ensure

generalizability by preventing the network from over-fitting (a scenario where the network learns to make predictions only on training data and fail to predict on unseen data). This layer typically takes place right after convolution layers. There are two common methods that are used to perform the pooling operation identified as average and maximum pooling. In average pooling, the average of all values within a pooling filter is passed on to the neuron in the next layer, where in maximum pooling, the maximum value within a pooling filter is only passed on to the neuron in the next layer. Figure 4.8 shows an example of average and maximum pooling operations.

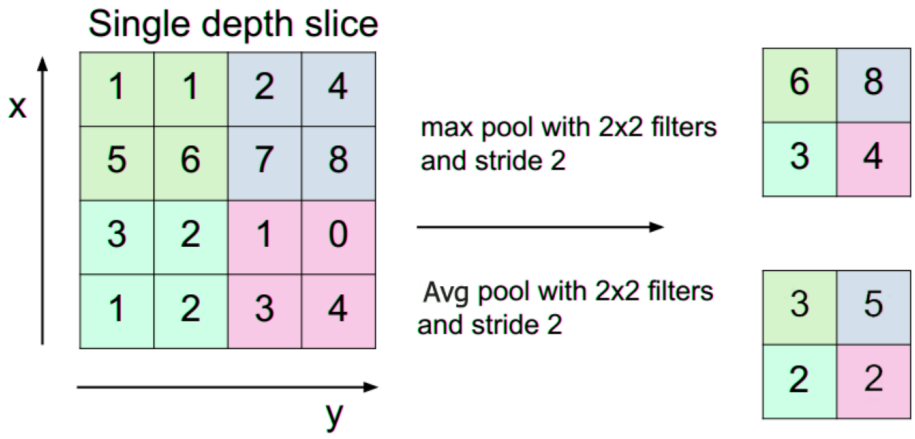


Figure 4.8: Average and maximum pooling output for 2×2 filter - adapted from [65]

Fully-connected layer usually comes as the last layer of the convolutional neural network. In this layer all neurons are fully connected with each other and that is why it is called fully-connected layer. This layer is similar to feed-forward neural networks. The benefit of fully connected layer is that unlike convolution layer, it learns features from all the combinations of features of the previous layer as shown in Figure 4.9, which is necessary to perform the regression or classification task.

4.1.3.1 Essential functional elements

Beside the layers of the convolutional neural network, the algorithm has other essential functional elements such as activation functions, which are used to apply a kind of non-linearity to the

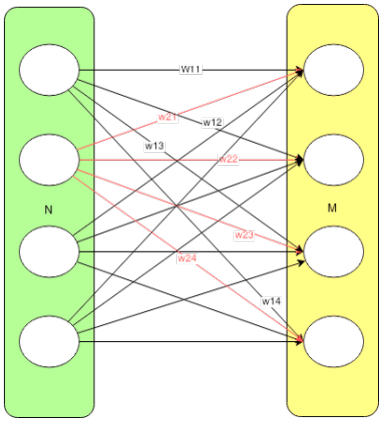


Figure 4.9: 4-node fully connected layer - adapted from [57]

outputs of the convolution operation in order to solve complex non-linear problems. For a classification task, the network produce outputs that are eventually classified into classes, to determine which class corresponds to which, a loss function is needed. Loss function computes the error or deviation between actual output and predicted output. The network is composed of parameters (weights and biases) that sit on the connections between the network neurons. To compute the proper value of these parameters, the error produced from the loss function is propagated all the way back through an algorithm referred to as the back-propagation algorithm. Using the back-propagation algorithm gradients (the change of network parameters with respect to the loss function value) are computed and hence the parameters of the network are tweaked or adjusted such that the network perform the required task (classification or regression) as needed.

1- Activation functions: there are many activation function types;however, we only highlight the most prominent ones used in convolutional neural networks.

- **Rectified linear unit (ReLU)**: this activation function is perhaps the most widely used activation function. This is simply a maximum operation against zero, where if the value is greater than zero then it is passed onto corresponding neuron, otherwise a zero is passed on (dead neuron).

- **Exponential linear unit (ELU)**: this is a variant of the ReLU activation and it is similar to it when applied to positive inputs. Unlike ReLU, negative inputs are not ignored in the ELU linearity, where an exponential function parameterized with an alpha parameter is applied to them instead of zeroing them.

- **Sigmoid**: is another form of activation that is different from the ReLU. The sigmoid function squashes the input values between 0 and 1 which bound activations and prevent them from blowing up. Sigmoid often causes what is called vanishing gradients because values are squashed all the time between 0 and 1.

- **Hyperbolic Tan (Tanh)**: is similar to the sigmoid function except that it expands the range of input values from 0 to 1 to 1 and -1. The gradient of the tanh is stronger than sigmoid, this means derivatives are steeper.

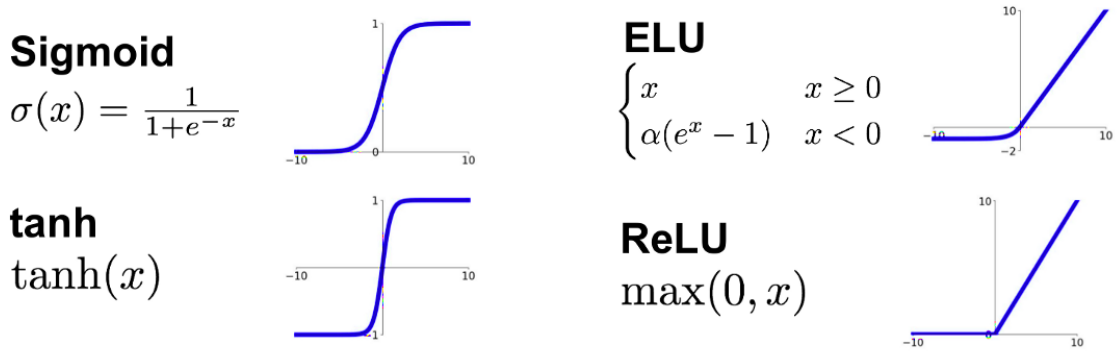


Figure 4.10: Example of common activation functions that are used in convolutional neural networks - adapted from [34]

2- Loss functions: there are various loss functions;however, we only highlight the most prominent ones used in regression and classification tasks.

- **SoftMax (Cross Entropy)**: this loss function is certainly the most widely used function in solving for classification tasks. Basically, the SoftMax classifier converts raw class scores

z_i of the nonlinearity in the preceding layer to a probability P_i between (0, 1) to bring the results to a common scale according to the this formula $P_i = \frac{e^{z_i}}{\Sigma_k e^{z_k}}$. The top probabilities from SoftMax classifier are then compared with actual labels of the available classes, hence evaluate the accuracy of the model.

- **Mean Squared Error (MSE)**: is commonly used in regression problems where it measures how much the predicted continuous value deviates from the expected or actual value as described in this formula $MSE = \frac{1}{n}\sum_{n}^{i-1}(Y_i - \hat{Y}_i)^2$. MSE is prone to outliers as it over weighs them, so usually mean absolute error (MAE) is used to handle data that contains a lot of outliers.

4.1.3.2 Common CNN architectures

In the past couple of years there has been unprecedented growth in the development of novel convolutional neural network architectures. The earliest developed CNN architecture known as LeNet-5 dates back to 1998. A list of common architectures that are used in nowadays image related problems are presented as follows.

- **AlexNet**: developed by Alex Krizhevsky in 2012 and won the ImageNet challenge [64] achieving a top-5 error of 15.3%, more than 10.8 percentage points lower than that of the runner up. The architecture comprises a total of five convolutional layers with three pooling layers and two full connected layers. AlexNet has about 60 million parameters and performs $\approx$ 1.1 billion MAC operations for one forward pass.

- **VGG**: the Visual geometry group (VGG) model was the winner of the ImageNet challenge in 2014. The deepest proposed model contained 19 convolutional layers, that is about 4x deep as AlexNet. Convolution filters of size 3×3 and 2×2 max-pooling ones were primarily used all across the architecture. This CNN architecture has about 138 million parameters and a single forward pass requires approximately 16 billion MAC operations.

- **GoogleNet**: this architecture is 22-layer deep and it won ImageNet challenge in 2015 with a top-5 error rate of 6.7%. The model has only 1.2 million parameters that is about 0.86% of VGG parameters. The massive reduction of parameters resulted into more complex architecture that employs what so-called Inception modules. An inception module is basically a network-in-network sub architecture that uses a 1×1 convolutional layer to reduce the number of input channels

- **ResNet**: this architecture is the deepest of all discussed architectures. Authors proposed variants with different number of layers (34, 50, 101, and 152 layers). The 152 layer deep architecture won the ImageNet challenge by achieving a top-5 error rate of 3.6%. Speaking of 152 layers means a hard training problem. To get over this problem, researchers included detours (residual connections and that is where it gets its name) around each batch of subsequent convolutional layers. This topology can be viewed as $y = F(x) + x$ where the network has to learn a residual function F(x) only. We utilize this type of CNN in our vision-based mass estimation work.

4.2 Modified nonlinear regression loss for sparse ground truth

Semi-supervised learning techniques such as self-training, mixture models, co-training, multi-view learning, and 3S-vector machines [87] are commonly used to solve sparsely annotated data problems. Essentially, input data is modeled somehow such that it is utilized to help with the prediction process via (e.g.) data clustering or self-teaching. In our proposed method [27], we take advantage of the response itself, where there exist a tractable relationship between each individual data point (input) and the ground truth (response). As a matter of fact, the ground truth represent the total sum of predictions of individual data points for a given period of time. Reinforcement learning with a Markov process assumption is one such example of sparse rewards and a tractable relationship where methods such as policy gradients can be employed. In a simpler case like this one, measurements can all be treated independently, and when predictions

from these measurements are aggregated with a summation to produce a total (accumulated) mass, a loss function for the predictive model can be realized.

The typical nonlinear regression formulation using mean squared error (MSE) loss can be slightly modified to incorporate an additional aggregation term over the predictions for each run (video) in k runs as shown in Equation 4.2. The only difference being that the predictions of image "x_{ij}" over a given video are summed and then compared to the ground truth and scaled by length "n" of the video. A run in this context represents a sequence/stream of images captured for a weighted amount of material. This type of simple modification to nonlinear regression opens up a major opportunity to bypass more costly or infeasible situations to obtain labels or ground truth in order to train a predictive model. The application presented in this work is one such situation where it is much easier to measure an accumulated mass as opposed to trying to obtain an individual mass for each measurement, which would be highly infeasible in any practical sense.

$$L(y; x; w) = \sum_{i=1}^{k} \frac{1}{n} \left(y_i - \Sigma_{j=1}^{n} f(x_{ij}; w) \right)^2 \tag{4.2}$$

4.3 Learning mass from images

Learning mass from images or mass flow from a video stream is possible in this problem context due to the constraints imposed on the system and the ability to learn complex patterns via end-to-end training with deep learning. More specifically, the camera has a fixed sampling frequency that is fast enough to measure all the material passing by, it is mounted at a fixed distance from the elevator, and the only additional factor needed is slat velocity to scale the accumulated mass. Intuitively, the velocity is scaling the mass to produce a mass flow, since if the mass was sitting still it would not be accumulated. Alternatively it can be thought of as a way to account for frame overlap.

Even though complex nonlinearities exist with the density and the material location in the image as shown in Figure 4.11 (e.g. the material further away is smaller in the image), these patterns can be learned by optimizing a deep neural network due to fixed locations between the camera

and the elevator. Understanding this reasoning about the system is crucial to designing and training a successful model in this and similar applications, since (e.g.) naively assuming counting pixels with material will correlate well with mass would lead to wasted efforts considering only a single factor such as image deformation.

Figure 4.11: Sample image from the bamboo dataset

The formulation to predict mass from images is shown in Equation 4.3, where f is a deep residual convolutional neural network that is parameterized by w. x_{ij} represents input image j in run i, and v_{ij} is elevator speed at time j in run i, and t is the frame per second of images.

$$Mass_{ij} = f(x_{ij}; w) \times v_{ij} \times t \tag{4.3}$$

To be able to predict mass using the deep convolutional neural network we use the modified loss function introduced in section 4.2 and only add the scaling factor $(v_{ij} \times t)$ to it to account for frame overlap as shown in Equation 4.4.

$$L_i(x, y; w) = \frac{1}{n_i}\{y_i - \sum_{j=1}^{n_i}(f(x_{ij}; w) \times v_{ij} \times t)\}^2 \tag{4.4}$$

The value of using images over volumetric methods will potentially be enhanced further when observing harvested materials of sugarcane or grains, since in the case of sugarcane the constituents such as billets and leaves (and therefore density) vary considerably more. The constituents and their sizes are observable in images though and therefore patterns more than

likely learnable by the same sparse ground truth methods. In grains the sizes of the kernels can change with moisture, and may exhibit other visual signs that correlate with density that an image-based algorithm could identify and learn to account for.

4.4 Temporal smoothing

Utilizing prior knowledge about the problem allows more custom formulation of the model, loss function, and training procedure to improve accuracy and stability. In this application it is clear that there is temporal correlation and images near in time should have more similarity in mass than images further away in time. To account for this during training, a regularizing term was added to the cost function with an associated hyper-parameter that allows penalizing a 1st-order lagged difference in the predicted mass values. Equation 4.5 shows full loss function for a run_i that includes prediction error and the additional temporal smoothing regularization term in red.

$$L_i(x, y; w) = \frac{1}{n_i}\{y_i - \sum_{j=1}^{n_i}(f(x_{ij}; w) \times v_{ij} \times t)\}^2 + \frac{\lambda}{n_i}\sum_{j=1}^{n_i}\{f(x_{ij}; w) - f(x_{i(j-1)}; w)\}^2 \tag{4.5}$$

Prediction is corrected by the elevator speed v and frame rate t $(7.5Hz)$ to account of frame overlap. Note that the gradient of the loss function has a sum of gradients in it, which if left un-normalized will amount to larger gradient updates for runs with more images even though the runs mostly contain the same total mass. Therefore, the loss is normalized by the number of images n in each run to equally weigh the gradient update from each run. The penalty strength is controlled by a hyper-parameter λ, and a λ of 0.05 resulted in the smallest average error after several experiments. We evaluated average error produced by several lambda values and noticed that higher values (> 0.1) aggressively penalized the loss function, resulting in higher error values. Binary search was applied to determine the best lambda value.

Although one might think that temporal smoothing can be achieved via using a recurrent neural network since RNNs are suited for sequential data processing; however, the very long sequences of runs would pose a challenge to RNNs. Moreover, it is worth noting that although we process stream of images (video), the data itself is not necessarily sequential and each image in the stream

is independent and representative of the information that is needed to be learned. The only relationship between the images is that they all together add up to a certain accumulated value. In our optimization problem, we do not just wish to predict the true accumulated value, rather we wish to predict the value of material content in each single image as accurately as possible. Therefore, there is no need to use an RNN/LSTM to process our data as it would only impose more complexity to the problem. Temporal smoothness is simply introduced to smooth the predictions between images near in time. This technique is meant to help the network learn faster by informing it that images near in time should have close predictions (mass content). Similarly, this kind of problem does not directly map to a recurrent neural network. Again, using an RNN would only impose unwanted complexity to the problem which is not favorable.

4.5 Gradient update

To perform a gradient update, we need to compare the prediction $\hat{y}_i$ or $f(x_{ij}; w)$ with the ground truth y_i, but the prediction and regularization terms contain sums in them, which means the gradient of the loss in Equation 4.5 does as well, as can be seen by Equation 4.6. Since the number of images is too large to fit on a single GPU vRAM (a problem for very sparse ground truth), we keep a running sum of the gradients, predictions, and regularizing term from each batch to greatly reduce memory requirements. The full gradient can then be calculated and applied at the end of the run using the accumulated terms from the gradient along with the ground truth, run length, and regularizing hyper-parameter. Equation 4.6 describes gradient of the full form of the loss equation associated with one run or run i. Terms in red are accumulated from each batch of a run and the full gradient calculated and applied when the end of a run is reached during the training loop.

$$\frac{\partial L_i}{\partial w} \leftarrow -\frac{2}{n_i}\Big[y_i - \sum_{j=1}^{n_i} \hat{y}_{ij}\Big] \times \sum_{j=1}^{n_i} \frac{\partial \hat{y}_{ij}}{\partial w} + \frac{2\lambda}{n_i} \sum_{j=1}^{n_i} \left\{ \Big[\hat{y}_{ij} - \hat{y}_{i(j-1)}\Big] \times \Big[\frac{\partial \hat{y}_{ij}}{\partial w} - \frac{\partial \hat{y}_{i(j-1)}}{\partial w}\Big] \right\} \tag{4.6}$$

Given the large and variable size of runs, we compute gradients and predictions in batches and accumulate them over the course of their respective runs as shown in the Pseudo-code in algorithm 1 .

Algorithm 1 Computing and applying gradients over a single epoch. With each run_i we have run length, images and speeds at the time of each image captured, and total ground truth mass

for `run` **in** `train_data` **do**
  y_{true} = **fetch_labels**(`labels`) ▷ Data loaded in chronological order
  for `batch, (images, speeds)` **in** `run` **do**
    x_b, v_b = **fetch_next_batch**(`images, speeds`) ▷ Data loaded in chronological order
    if `New_run` **then**
      `run_remainder` $\leftarrow$ `mod(run_length, sizeof(batch))`
      `iterations` $\leftarrow$ `ceil(div(run_length, sizeof(batch)))`
      `New_run` $\leftarrow$ `False`
    if `iterations > 1` **then**
      $\hat{y}_b \leftarrow f(x_b; w)$ ▷ Predict using DNN
      $\hat{y}_{b_{vt}} \leftarrow \hat{y}_b \times v_b \times t$ ▷ Correct frame-overlap
      $\hat{y}$ += $\Sigma\hat{y}_{b_{vt}}$, $\hat{y}_{b_{grad}}$ += $\Sigma\frac{\partial\hat{y}_{b_{vt}}}{\partial w}$ ▷ Accumulate predictions and gradients
      $\hat{y}_{smooth}$ += $\sum\{(\hat{y}_{b_{vt}} - \hat{y}_{(b_{vt}-1)}) \odot (\frac{\partial\hat{y}_{b_{vt}}}{\partial w} - \frac{\partial\hat{y}_{(b_{vt}-1)}}{\partial w})\}$ ▷ Accumulate penalty
      `iterations` $-= 1$
    else
      if `batch` **contains** `curren_run_images_ONLY` **then**
        $\frac{\partial L_i}{\partial w} \leftarrow \frac{-2}{n_i}(y_{true} - \hat{y}) \odot \hat{y}_{b_{grad}} \oplus \frac{2\lambda}{n_i}\hat{y}_{smooth}$ ▷ Apply gradients
        $w \leftarrow w + \alpha\frac{\partial L_i}{\partial w}$ ▷ Update weights
        `New_run` $\leftarrow$ `True`
      else
        Do `steps 17` $\rightarrow$ `18` **for** `images` **in** `current_run`
        Do `steps 4` $\rightarrow$ `14` **for** `images` **in** `next_run`

4.6 DNN model architecture and training procedure

We consider complexity and size as essential factors in the developed architecture, where eventual application goes on embedded hardware at mass scale and saving every bit of computation to a minimum is highly desirable to save costs. To assess the complexity of our dataset we adopt a residual like style [28] and follow a systematic approach in DNN design, starting shallow then reproducing accordingly. A 9-layer DNN with exponential linear units (ELUs) and residual connections, defined as – "RES9-ER" shown in Figure 4.12, was found to have good predictive accuracy as well as fast training and inference.

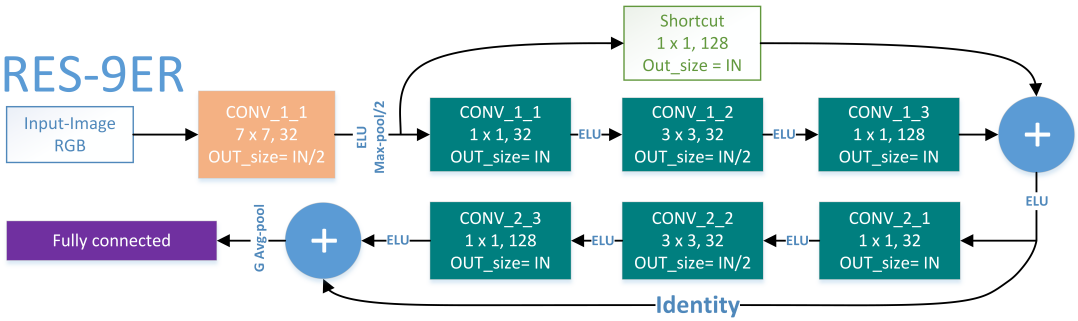


Figure 4.12: Reduced residual 9 architecture (RES-9ER)

Similar models with rectified linear unit (ReLU) activation and 16 layers were also investigated before converging on this final architecture. ELUs activation were considered in our DNN because it showed better noise dampening and more stable signal as well as helped converge faster than ReLUs. Figure 4.13 shows the training loss decay when using ReLU activation functions (in light blue) versus when using ELU activation functions (in orange). As shown, ELU-based loss signal converges faster than ReLU-based loss signal.

The 16-layer network – "RES-16E", barely out-performed the 9-layer network – "RES-9E". For this reason, we investigated the learned features in every layer of RES-9E and observed redundant features, hence we reduced the number of filters in RES-9E to introduce reduced RES-9E, the best performing architecture defined as – "RES-9ER". Our observation of the redundant features is empirical and validated using this work [18]. Figure 4.14 shows a selected feature 64, in left

column of the figure against a set of other similar features 1, 14, 62, in the middle column, whereas regions of different pixel information are highlighted in pink in the similarity map in the right column in the figure.

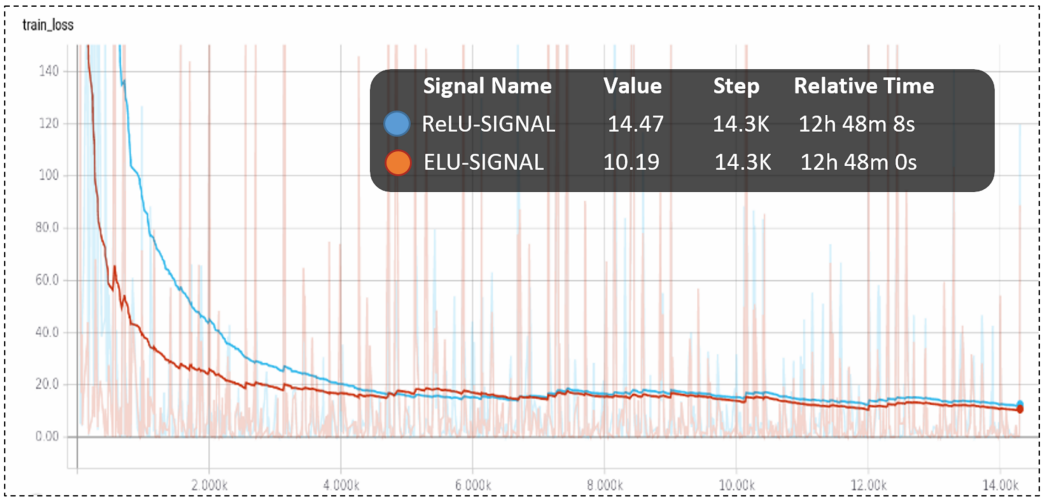


Figure 4.13: Training loss decay when using ReLU activation vs. ELU activation

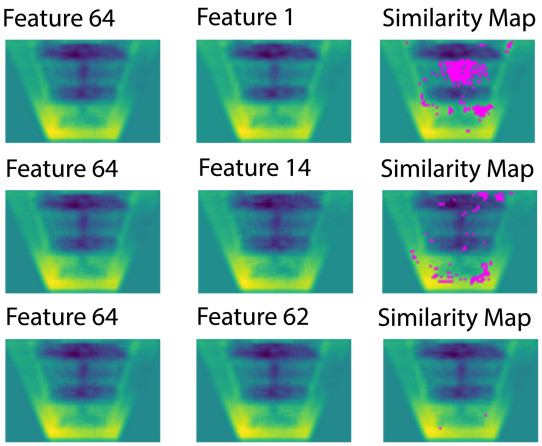


Figure 4.14: Illustration of redundant feature maps. This type of investigation helped inform of a suitable architecture that balanced accuracy, generalization, and stability

RES-9ER inference time for a batch size of 8 is ≈348FPS (frame size, fifth 720p) when running on 1080-Ti GPU, and ≈91FPS when running on Intel Core-i7-7600U CPU. The FPS is proportional

to batch size (i.e. sampling frequency) of images and subject to I/O or memory bounds. Table 4.1 summarizes the sizes of RES-9E, RES-16E, and RES-9ER.

Table 4.1: Summary of developed architectures and their sizes

Architecture	Number of trainable parameters	Number of layers
RES-16E	959489	16
RES-9E	154113	9
RES-9ER	45921	9

4.7 Results and analysis

4.7.1 Visual explanation of DNN functioning

To get an intuition on what the network actually learns and what parts of the image contribute the most to the outcome, we follow a generalized gradient-based CNN visualization approach to visualize the learned mass in images. We adapt the grad-cam method proposed by this work [11] and develop an implementation that is compatible with eager mode in Tensorflow [75].
Figure 4.15, Figure 4.16, and Figure 4.17 show three different images and their corresponding grad-cam heat-maps. The heat-map colors indicate the parts of the image that contributed the most to the outcome being the dark orange representing the highest contribution (dark orange contributes more to the output). As shown in Figure 4.15, the network learns when there is not bamboo present in the image and the network learns when there are various amounts of bamboo in the image as shown by the grad-cam in Figure 4.16 and Figure 4.17.

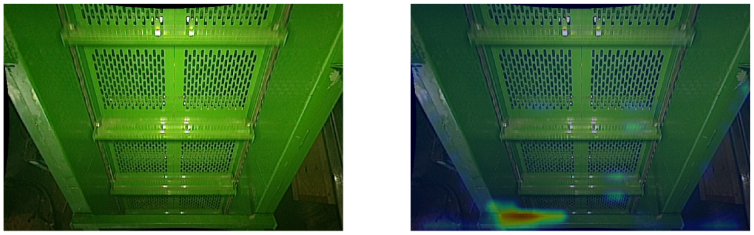


Figure 4.15: No flow of material

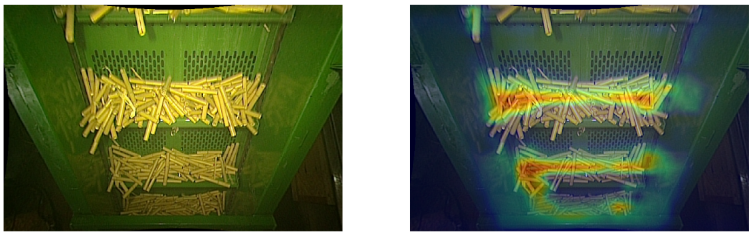


Figure 4.16: Medium flow of material

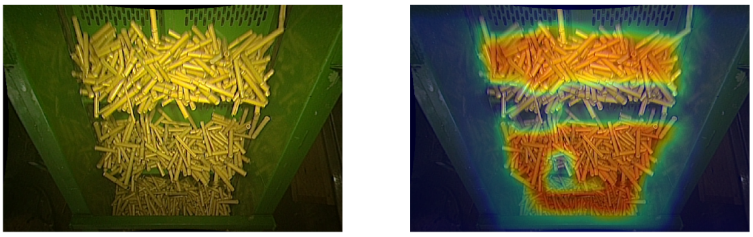


Figure 4.17: High flow of material

Figure 4.18 shows a snapshot of a continuous run of running bamboo on a sugarcane elevator (image to the left) and its live Grad-CAM visualization (image in the middle) with signal plot (image to the right) showing current prediction of material in pounds and the accumulated prediction under the curve. Ground Truth (576lbs) represents the total ground truth mass for a complete run. You can view the live prediction of running bamboo using this link LIVECAM [26].

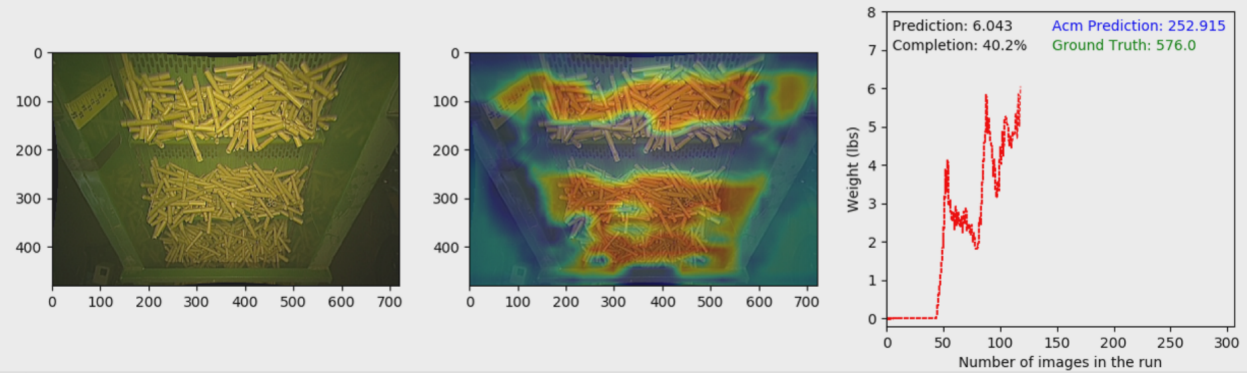


Figure 4.18: Snapshot of a live Gradcam heatmap of a random frame

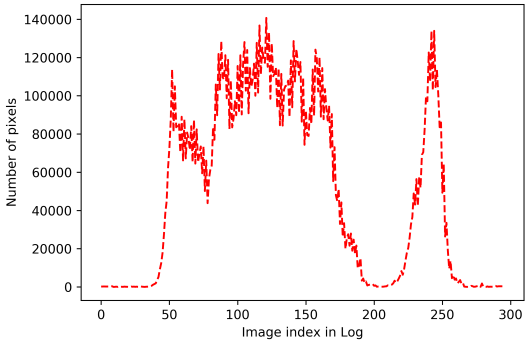
4.7.2 Temporal smoothness effect

Training with temporal smoothness for the different developed architectures reduced the overall error by 0.6% on average. Table 4.2 lists the average errors obtained for RES-9E, RES-16E, and RES-9ER with and without temporal smoothness.Thus far, even without temporal smoothing the signal had very good stability which is much better than the volume-based signal. For a select run (RUNX) from the test set that has a ground truth of **261.3KG**, the volumetric based approach predicted a total mass of 279.6KG which makes a percent error of 7.1%, while the DNN based approach with and without temporal smoothness made a prediction of 269.8KG, 262.8KG respectively, which translates into percent errors of 0.5% with smoothness and 3.3% without smoothness. Using temporal smoothness improved error by 6.6% compared to te volume based prediction and 2.8% compared to the vision-based (DNN) prediction using ELU activation without temporal smoothness.

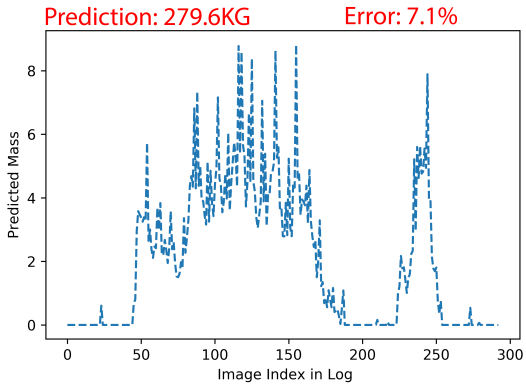
Table 4.2: Summary of architectures average error

Architecture	Test-set average error	
	W/O Temporal Smoothness	W/ Temporal Smoothness
RES-16E	5.2%	4.7%
RES-9E	5.4%	5.1%
RES-9ER	**5.5%** (±0.18)	**4.5%** (±0.07)

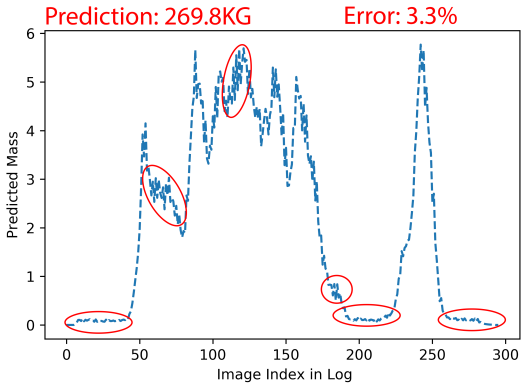
For visualization purposes and to assess where in the signal the temporal smoothness made changes to the prediction signal, we plot the prediction signal obtained by the DNN with and without applying temporal smoothness for RUNX. Also, to show consistency and smoothness of the vision-based signals, we plot the volumetric equivalent. Lastly, we plot RUNX signal based on counting the number of pixels in each frame. Counting pixels method does not generalize, i.e. has to be obtained for each run separately; however the sole purpose of using this method is to showcase the consistency of the vision-based signal of RUNX with the pixel based signal. Visualization of the aforementioned plots is shown in Figure 4.20.



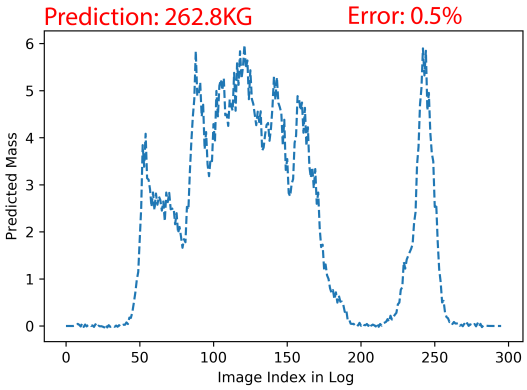
(a) RUNX signal shape plotted based on counting the pixels of bamboo in each individual frame



(b) Predicted signal of RUNX based on the volumetric algorithm



(c) Predicted signal of RUNX based on the DNN algorithm without applying temporal smoothness. Regions circled in red are the parts of the signal that show observable difference when compared to the signal with temporal smoothness applied



(d) DNN or vision based predicted signal with temporal smoothness applied. We can see the predicted signal shape of RUNX based on the DNN approach is relatively consistent with the signal shape obtained by counting pixels in each frame

Figure 4.20: Plotting RUNX signal using different methods (pixels based, volumetric based, and vision based) to showcase the overall shape of the signal

4.7.3 Comparison to volume-based predictions

To demonstrate the robustness of the vision-based signal, we compare the mass predicted signal using RES-9ER against volume-based mass predicted signal using stereo camera in the following scenarios: 1) Incremental and decremental variable flow - Figure 4.21, 2) Intermittent flow - Figure 4.22, and 3) Poor lighting conditions - Figure 4.23.

The vision-based prediction outperforms the volume-based prediction in each of the scenarios which shows the robustness of the algorithm when posed to variable environmental conditions as opposed to the volume-based signal. We would like to note that the run selected for the poor lighting conditions is one of the runs identified to have too low lighting to use volume estimates on.

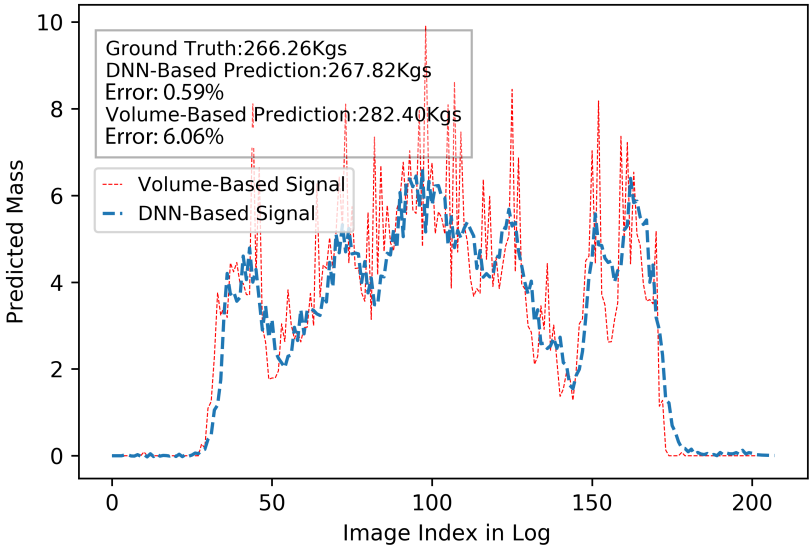


Figure 4.21: Overlay of volumetric and vision-based signals showing incremental/decremental mass flow

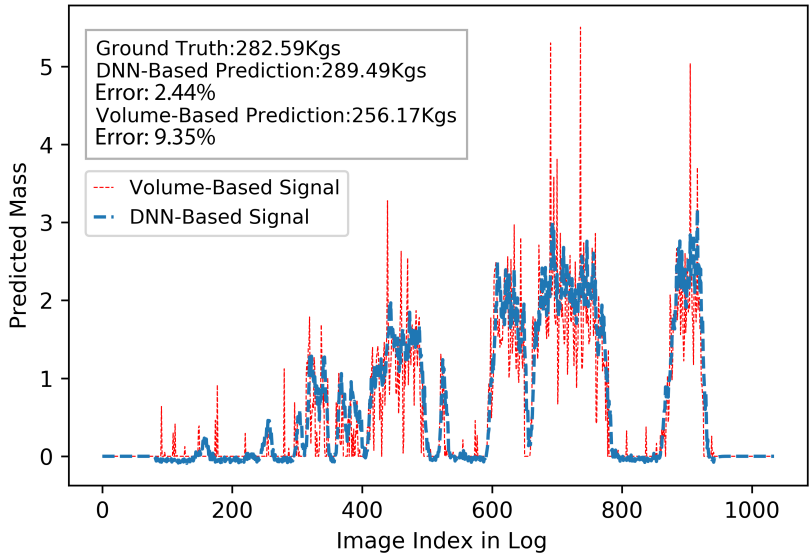


Figure 4.22: Overlay of volumetric and vision-based signals showing intermittent mass flow

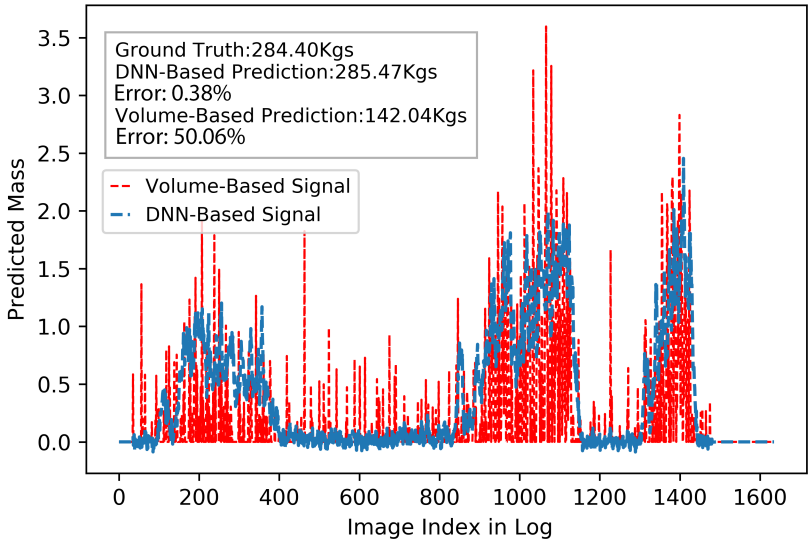


Figure 4.23: Overlay of volumetric and vision-based signals of mass flow under poor lighting conditions

4.7.4 Error distribution and outliers

Studying error distribution of the dataset helped us understand where the DNN algorithm made poor predictions as well as how to improve generalization. No extreme outliers were found in the validation or test sets, but the training set contained a single outlying run circled in red in Figure 4.25. The total mass prediction of the outlying run was too low ($\sim 40\%$ of what it should have been), but yet the DNN mass prediction per image before scaling with elevator speeds and capture time was very high. This was also confirmed by a grad-cam analysis as seen in Figure 4.24, where even the sidewalls are being incorrectly incorporated into the prediction of mass. Therefore, we investigated the speed signal and it was found that the outlying run had a very low elevator speed (average of 0.1 m s^{-1}) with large variations in speed.

It is likely that the optimization process tried to use the reflections of bamboo on the side of elevator to compensate for the low total predicted mass (which was actually a problem of low fidelity from the speed sensor at low elevator speeds). Since this outlier was an extreme corner condition of very low elevator speed that is not at all likely in real applications, thus it could be removed and the network retrained to likely achieve better overall results. Similarly, a corrective action could entail using a speed sensor with higher resolution and fidelity over a broad speed

range. Figure 4.25 shows histograms of error distribution from the DNN and volume based predictions across the entire dataset. The vision (DNN) method can be seen to perform much better even though we included a large outlier due to elevator speed abnormalities, and also gave the volume estimates an unfair advantage by excluding low-light runs for which it faired very poorly.

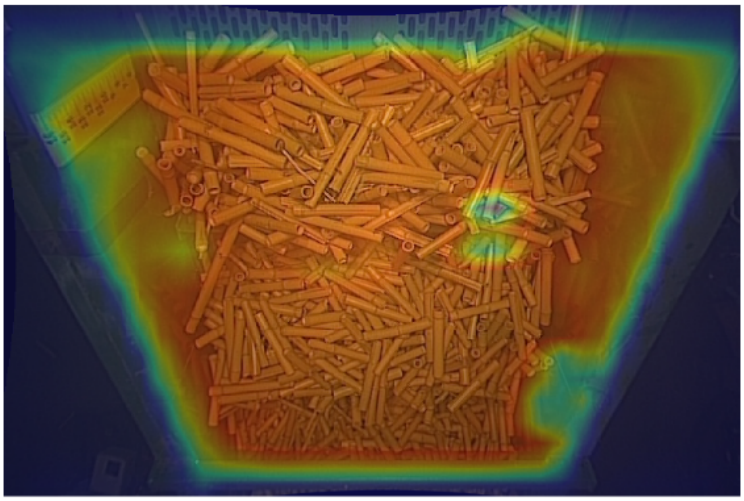


Figure 4.24: Gradcam visualization of an image from the outlying run

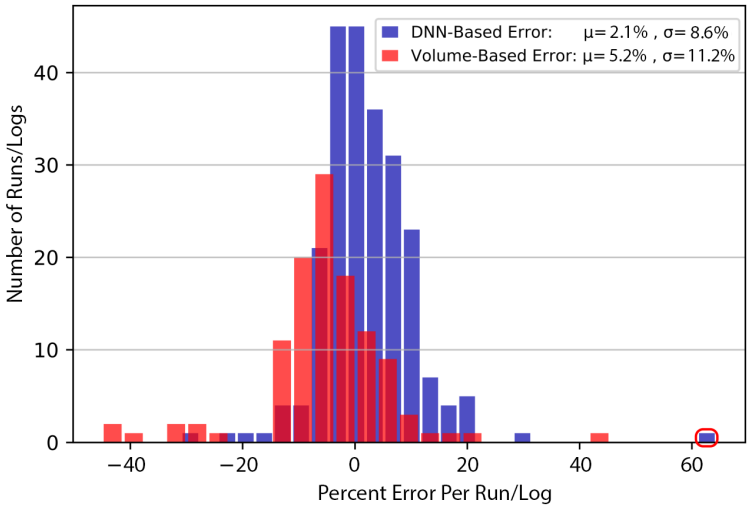


Figure 4.25: Error distribution of predictions by the DNN and the volume algorithms. **Note**, runs with poor lighting are excluded for the volumetric algorithm

Results summary

- ✓ Developed a DNN algorithm that can accurately capture a physical property (mass) of material.
- ✓ A dynamic algorithm was devised to accommodate the memory limitations of GPU working with large sequences of images.
- ✓ Mass was estimated using images with average error of 4.5%.
- ✓ A temporal smoothness term was devised to improve the accuracy of predicted signals.
- ✓ A visualization technique has been introduced to investigate what the DNN algorithm learns behind the scenes.

4.8 Conclusion

In this work [PHASE II] we proposed a semi-supervised algorithm that makes inference on mass flow of material from sequences of images by training a deep neural network with sparse ground truth, and showed improvements over older and more expensive methods that must first acquire a volumetric estimate of the material and then calibrate the density. With the careful DNN design, an average error of 4.5%, better signal shape, and high frame rate up to ≈348FPS were obtained. The results obtained herein support applying the methods derived to predict mass flow of field harvested sugarcane as measured on a machine [PHASE IV], which experiences many other complicating factors affecting density and visual appearance of the material.

CHAPTER 5. VOLUMETRIC-BASED MASS FLOW ESTIMATION UNDER REAL ENVIRONMENT OPERATION

In this chapter we describe a volumetric-based approach to estimate mass flow of material in a real operating environment (on real sugarcane harvesters). We leverage the proposed methods in chapter 3 (phase I), i.e. 3d point cloud generation of volume, and apply it to real world data. In chapter 4 (phase II) we proved that the vision-based approach outperforms the volumetric based approach for mass estimation; however, we still use the volumetric based approach to explore its performance on real world data as well as to form our baseline comparison when applying the vision-based approach on field data.

5.1 Materials and methods

5.1.1 Fundamentals of operation

The fundamentals of operation presented in this chapter are essentially the same as in chapter 3; however, the operation of the yield monitor discussed herein consists of measuring the volume of the sugarcane billets instead of bamboo on the machine elevator during harvesting, and converting the measured volume to a mass via a calibrated density. A stereo camera operating at 7.5Hz was used to produce a dense 3d point cloud of the material.

Laboratory testing (phase I) showed highly consistent results with the stereo cameras as long as lighting was maintained above a certain level, which was easily achieved with supplementary LED lighting. The caveat then for a typical volume-based yield monitor is the density of the material required to translate to mass. Grains are known in moisture sensing to expand and contract with moisture, and change both the particle and bulk densities [24]. In the case of highly non-uniform shape characteristics like sugarcane billets, which are long slender rods, bulk density has the

potential to be more complex from the shape factor alone, along with particle density expected to additionally contribute to density complexity.

Mass flow up to a scaling constant depending on the time period to estimate over is shown in Equation 5.1. The stereo camera estimates the volume within the region of interest (ROI), which is denoted as V_c and has units of meters cubed per meter up to a scaling constant. Then this quantity is scaled by the distance the elevator moves ($\Delta t \times V_e$) in between the next volume estimate to find an incremental accumulated volume. A simplifying assumption inherent in this formulation is that the volume calculated (V_c) is spread evenly across the ROI, since the incremental accumulated volume V_Δ is directly proportional with the distance the elevator moves. The incremental volume is then converted to mass via a multiplier (calibration factor/density) as seen in Equation 3.1. Any error in density of the material can be seen to directly contribute to error in predictions of mass, and therefore yield as shown in Equation 5.2.

$$V_\Delta = \Delta t \times V_e \times V_c; \quad m_\Delta = V_\Delta \times \rho \tag{5.1}$$

where:

V_Δ = incremental accumulated volume (m^3)

V_c = proportional to cross sectional area of material on the elevator from stereo camera (m^2)

Δt = image capture time (s)

V_e = harvester's elevator velocity level (m s^{-1})

m_Δ = incremental accumulated mass (kg)

ρ = density (kg m^{-3})

$$Yield = \frac{\dot{m}}{w \times v_m} \tag{5.2}$$

where:

$Yield$ = (Mg ha^{-1})

$\dot{m}$ = mass flow (kg s^{-1})

V_m = vehicle speed (km h^{-1})

w = row width (m)

To convert volume measurements to mass, an algorithm was devised and fit that explicitly uses the incremental volume to predict the density, and then use it as a multiplier on the volume to convert to mass as shown in Equation 5.3

$$Mass = f(max(V_c - \beta, 0); \theta) \times V_e \times \Delta t \tag{5.3}$$

Where f is a feed-forward neural network parameterized by θ that outputs a prediction of mass based on raw volume V_c, scaled by elevator speed V_e and capture time Δt. Note that the underlying density is implicitly estimated. The neural network is composed of 4-layers with a total of 256-hidden units as shown in Figure 5.1. The β-parameter is also fit to account for any positive bias present since the stereo volume calculation was designed towards not missing any volume. This was also a meaningful formulation since if the β parameter turned out to be negative then it would indicate a volume estimation is in need of refinement, since it would be compensating for volume not detected.

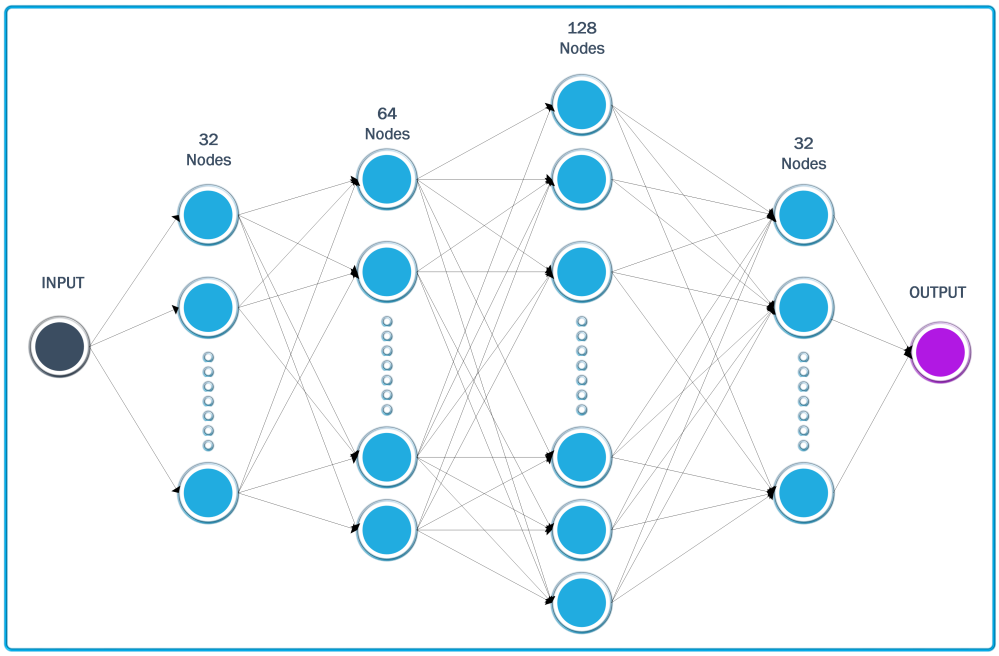


Figure 5.1: Feed forward neural network used to estimate density to convert volume measurements into mass

5.1.2 Field setup and data summary

Field data was collected in the course of 3 consecutive years (2014 through 2016) and in 4 different regions (Brazil, Florida, Louisiana, and Texas) to ensure robustness of the system to various environmental factors that could influence bulk and particle densities of the material. Data is comprised of 1567 runs (over 3M frames), and is split between burnt and green cane. This was an important part of the design of experiment since burnt cane, in which the leaves and fibrous trash were burnt off, was projected to represent the high end for density, whereas green cane was expected to vary more depending on the amount of trash and ability of the primary extractor fan to remove trash. Table 5.1 summarizes runs distribution based on location, season of harvest, and type of sugarcane.

Table 5.1: Runs distribution of years, region of harvest, and material type

Crop Type	Region	Sugarcane Harvest Year			Total/ Region
		2014	2015	2016	
	Louisiana		669		669
Green	Brazil	166			166
	Florida			264	264
Burnt	Texas	66	79		145
	Florida			323	323
Total		232	748	587	**1567**

The laboratory and field systems are identical except for the wireless transceivers that were used to collect ground truth from scales, which were installed on the wagons as shown in the diagram in Figure 5.2. The wagons were typically six or nine metric ton capacities. An image processing unit (stereo camera + algorithm) was used to generate colored images and a 3d point cloud (converted into volume) of material. A speed sensor was used to capture the elevator speed. The image processing unit was controlled via CAN bus and data was transferred to a dedicated stereo logger via an Ethernet link between the image processing unit and the stereo logger.
Early on in testing (2014, and Texas 2015), ground speed and fan speed were used to induce mass flow changes as well as changes in material composition (and therefore density). While they are

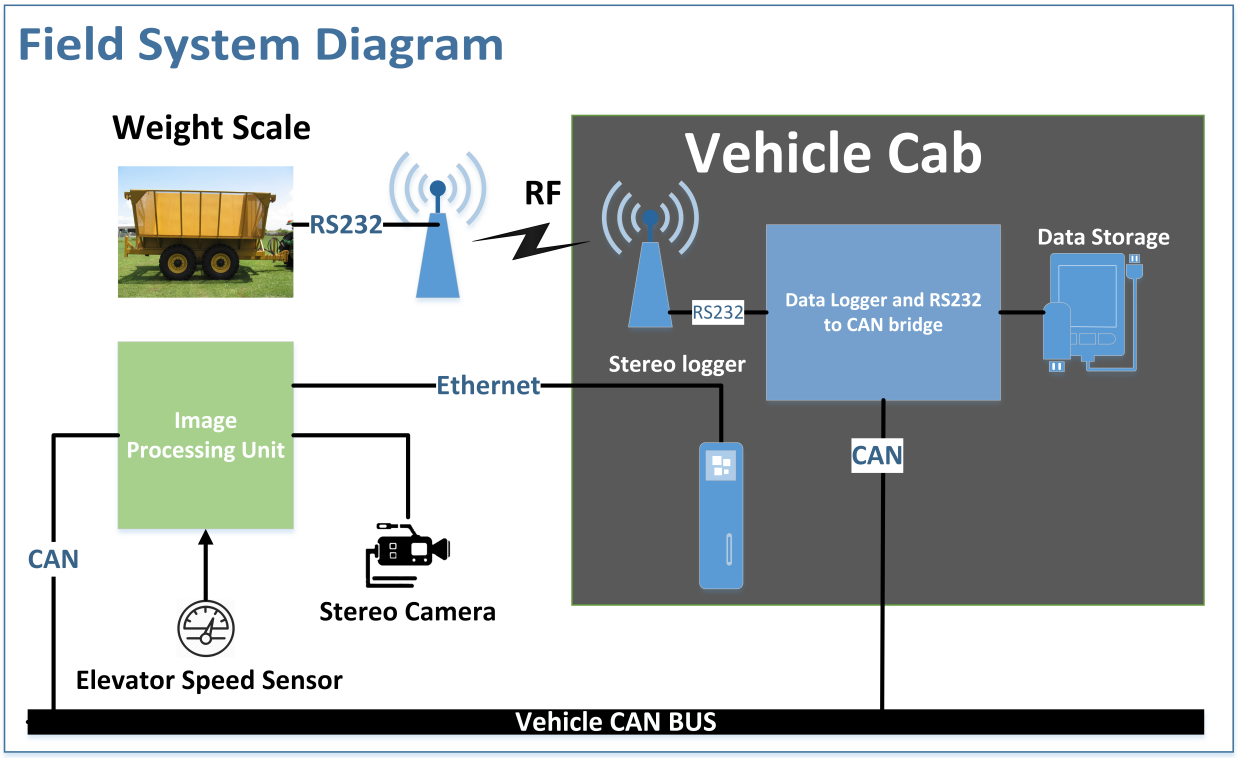


Figure 5.2: Block diagram of the field system setup highlighting the components used for data generation and logging

useful for obtaining more variation in the dataset, they are not globally consistent factors since the magnitude of the effect of ground speed on mass flow and fan speed on material composition depends on yield and trash levels, respectively. Mass flow, ground speed, and fan speed from field runs are summarized by histograms as shown in Figure 5.3. Note run, log, and video refer to a sequence of frames with an overall accumulated mass.

5.1.3 Potential failure modes

The stereo camera system worked well even with very little maintenance, up to and including when the lens on the camera were severely covered with dirt or dust. However, there were a few rare instances when a perfect storm occurs and external factors affect the camera performance. These conditions are hard to remedy, but fortunately as mentioned were found to be rare. An

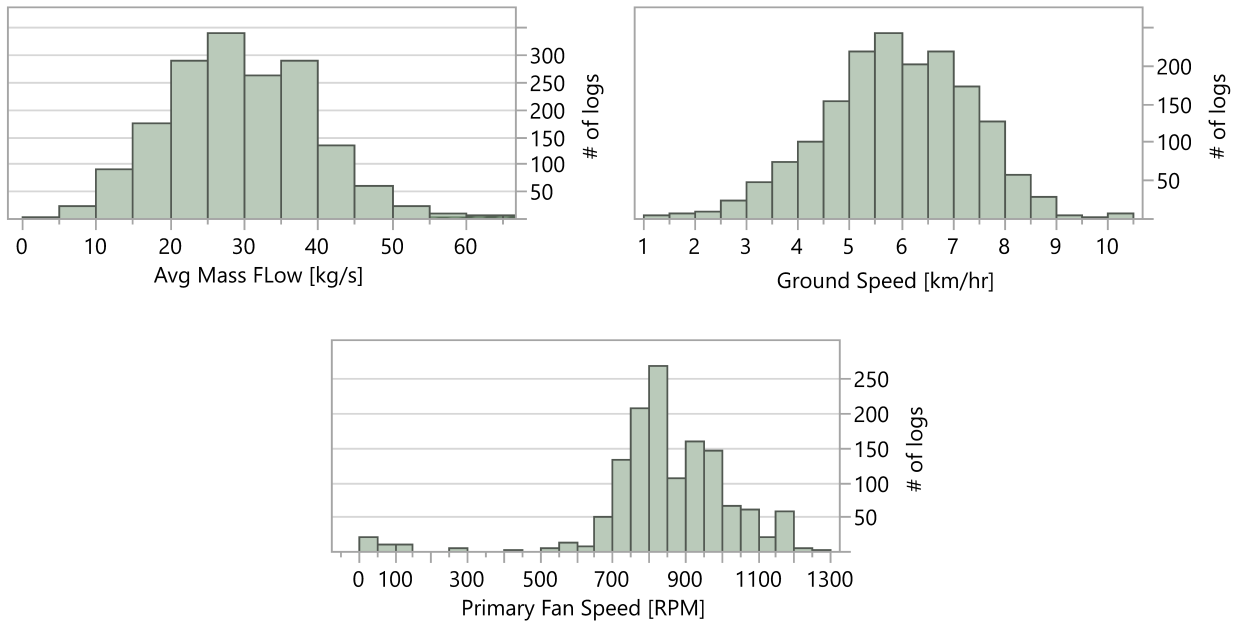


Figure 5.3: Summary of data from testing runs showing range of factors covered - not including empty runs

instance was found for a short period of a day when harvester was driven in a particular direction, the sun would shine directly into the camera and wash out a portion of the point cloud as shown in Figure 5.5. Another issue that occurred is when material flows were very high and the trash, especially dry and light, could fly up and block the camera view as shown in Figure 5.6. Again this was a relatively rare condition, but had the potential to increase density for those runs since volume estimates for those point clouds were biased low.

5.2 Results and analysis

The volumetric approach was evaluated using two methods: 1) box-cox (power) transform, and 2) feed-forward neural network that was described in Equation 5.3. The first approach is less complex, however it does not perform as good as the second one. We study the first approach as it offers some insight into the data and helps us understand how it can be improved.

Figure 5.5 Sugarcane under direct sunlight

Figure 5.6 Trash blocking the view of the camera

Figure 5.7: Left: Sunlight washing out portion of the image. Right: Trash flying up and blocking the view of the camera, hence affecting the point cloud estimation

The benefit of applying a transformation to the raw volumetric data can be seen in Figure 5.11. In an ideal situation, the increase of volume flow should not affect density, i.e. density should not decrease as volume flow increase. However, this is not the case in the used system as there exist some variations in the density values. Applying a box-cox transformation (xfm) with a lambda value of 0.3115 (which was found by sweeping over a range of values and finding the lambda value that minimizes the variations in the density values the most) to volume measurements prior to estimating density was found to improve mass estimates through minimizing variation in density values. We notice after applying box cox transformation to the raw volumetric data that the decreasing density trends observed in Figure 5.9 have mostly disappeared as can be seen in Figure 5.10.

To absorb the variation observed with the estimated mass based on estimating density, we consider a different approach where we use a feed-forward neural network that directly estimates mass from raw volume. Based on Equation 5.3, raw volume is fed into the neural network and a mass per meter value is estimated. Performance of the neural network and box-cox transformation approaches is evaluated by comparing the results of both methods.

When using a neural network, the data is split into train, validation, and test sets. The neural network is trained on the training set and performance is evaluated on the testing set. The

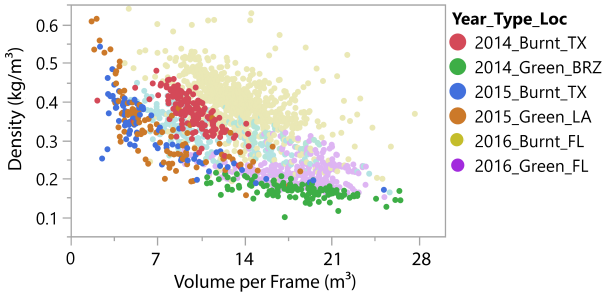


Figure 5.9 Density Vs. Volume

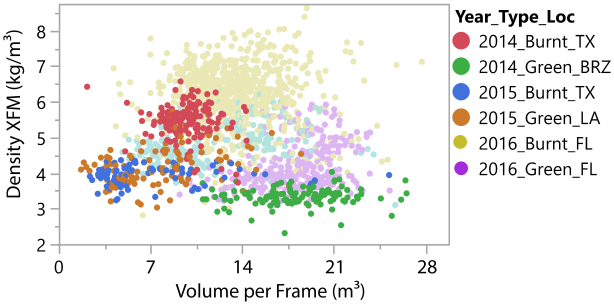


Figure 5.10 Density (with box-cox xfm) Vs. Volume

Figure 5.11: Left: Before transformation of volume, clear trends can be seen of density decreasing with volume flows. Right: After transformation of the volume to account for bulk density changes in material with increasing flow rates. The trends disappear, showing the phenomenon has been adjusted for

validation set is used to monitor the training progress to avoid a common problem in machine learning known as over-fitting where the neural network only learns the details of train data and becomes incapable of learning from unseen data. Table 5.2 summaries the test-set average error obtained for each region as well as for all-fields combined for both transformed volume and neural network based estimation. From Table 5.2 we observe that the error of both the transformed

Table 5.2: Test-set average error per region.

	Test-set average error	
Region	**Transformed volume estimation**	**Neural Network estimation**
Louisiana	12.23%	10.55%
Texas	8.31%	**7.78%**
Brazil	9.26%	8.50%
Florida	18.70%	15.78%
All-Fields	30.39%	25.12%

volume and neural network based estimation is comparable when performed on each region independently; however, the transformed volume performs poorly when combining all fields together. Although the neural network error is not quite low, it is still lower than the transformed volume by $\approx 5\%$. This shows how the neural network can learn better when non-linear behavior

is introduced. The box-cox transformation works best when data is split by year, season, and region of harvest as this obviously reduces the variations in estimated density values.

Florida shows high average error compared to other fields, and that is due to the mix between green and burnt cane, which obviously have varying densities. Louisiana's average error comes after Florida's due to a shift in the density values that occurred during harvesting. It was found that the density changed around a time when material consistency degraded due to operations and environment, and more dirt and root balls showed up in point clouds. Hypothetically, covering the material in dirt could easily increase density and explain the fluctuations and overall net increase in density experienced there.

To get a closer look at the performance of neural network and transformed volume estimations on field data, we overlay histograms of error for both types of estimation as shown in Figure 5.12. We notice that the transformed volume method is skewed, where it over-estimates mass on several runs, while the neural network approach maintains much more balanced estimates.

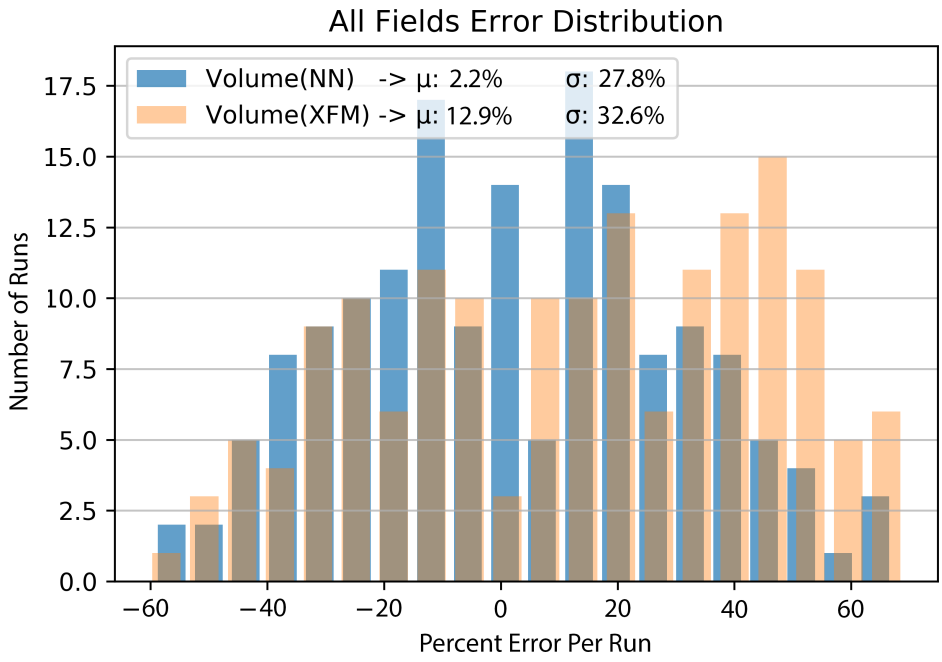


Figure 5.12: Histogram overlay of error distribution of neural network (NN) and transformed volume (XFM) estimates of all-fields test set

5.3 Conclusion

In this chapter we described two approaches to estimate mass based on the volumetric surface of material on the elevator of a sugarcane harvester using a stereo camera. The first approach (volumetric transformation) is contingent on a calibrated value, or density, to convert from volume to mass. The calibration values are relatively easy to obtain by tracking volume into a semi and getting a weight back from the mill. The second approach (neural network) directly estimates mass values. When separated out by season, region, and crop type the average error is quite reasonable, reaching 7.78% when estimated via neural network. Finally, this work serves as an excellent baseline comparison for solving the same problem using visual processing (images) via a deep neural network.

CHAPTER 6. VISION-BASED MASS FLOW ESTIMATION UNDER REAL ENVIRONMENT OPERATION

In chapter 4 we discussed the possibility of estimating mass via an end-to-end deep neural network from a sequence of images. We devised a set of methods that were applied to the bamboo dataset which was collected under controlled testing environment and achieved an average error of 4.5%. In this chapter we wish to transfer the developed methods in chapter 4 and use them to estimate mass of flowing sugarcane on a sugarcane elevator under real operation environment. The sugarcane dataset is a lot bigger and more complex than the bamboo dataset, so we seek to explore to what extent we can apply transfer learning [77] (a learning paradigm in deep learning that aims to transfer learned knowledge from one application to another) with minimal to no changes to the devised methods. This chapter also investigates if the vision approach would still outperform volumetric estimation under real environment operation just like how it did under laboratory testing.

6.1 Sugarcane dataset acquisition

6.1.1 Dataset summary

The sugarcane dataset has a total of 1567 logs/runs (over 3M images) acquired from four different regions (Texas, Brazil, Louisiana, and Florida) over 3 consecutive years from 2014 up until 2016. The dataset is split between burnt and green cane and a summary of the dataset is shown in Table 6.1 (same as Table 5.1).

Table 6.1: Runs distribution of years, region of harvest, and material type

Crop Type	Region	Sugarcane Harvest Year			Total/ Region
		2014	2015	2016	
	Louisiana		669		669
Green	Brazil	166			166
	Florida			264	264
Burnt	Texas	66	79		145
	Florida			323	323
Total		232	748	587	**1567**

6.1.2 Dataset pre-processing

The sugarcane dataset has undergone undeniable pre-processing stage to get the data prepared to be processed by the deep learning model. As noted in chapter 3, images were collected from the same stereo camera that was used to produce the point cloud for volumetric estimation.
Nearly 200TB of unprocessed (raw) log files were acquired and undergone several stages of data pre-processing that can be summarized as follows: 1) Search algorithm that was used to look up target logs/runs from the server end. 2) Batch-based data acquisition of log files from the server end to local drives. 3) ID files generation for each raw log file. 4) Log trimming, where data is extracted based on given start and end timestamps of logging from each raw file. 5) Meta-data extraction from trimmed logs. 6) Trimmed logs translation into images and metadata that are eventually used in the deep neural network. Figure 6.1 shows the overall process of data acquisition until image extraction.

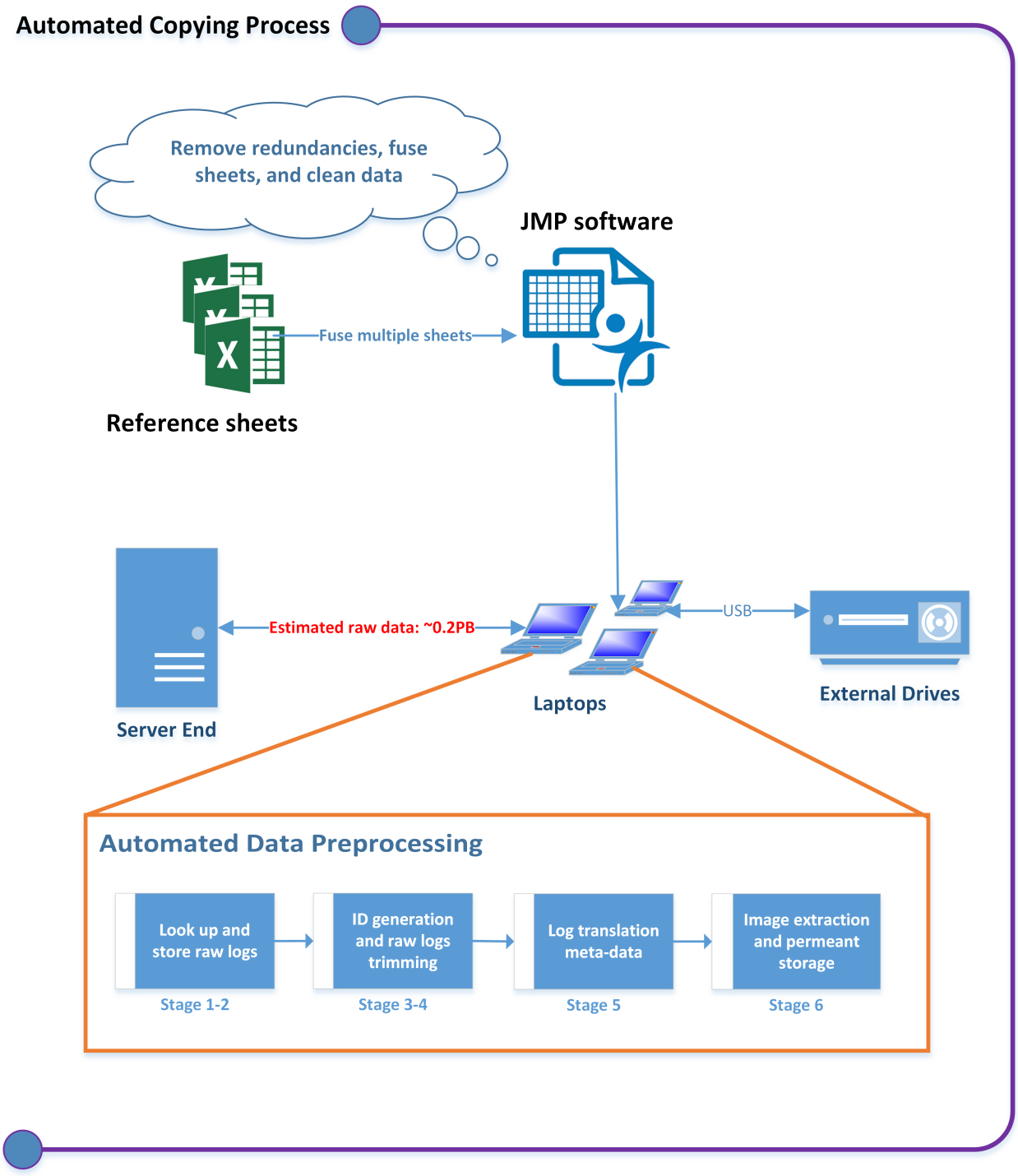


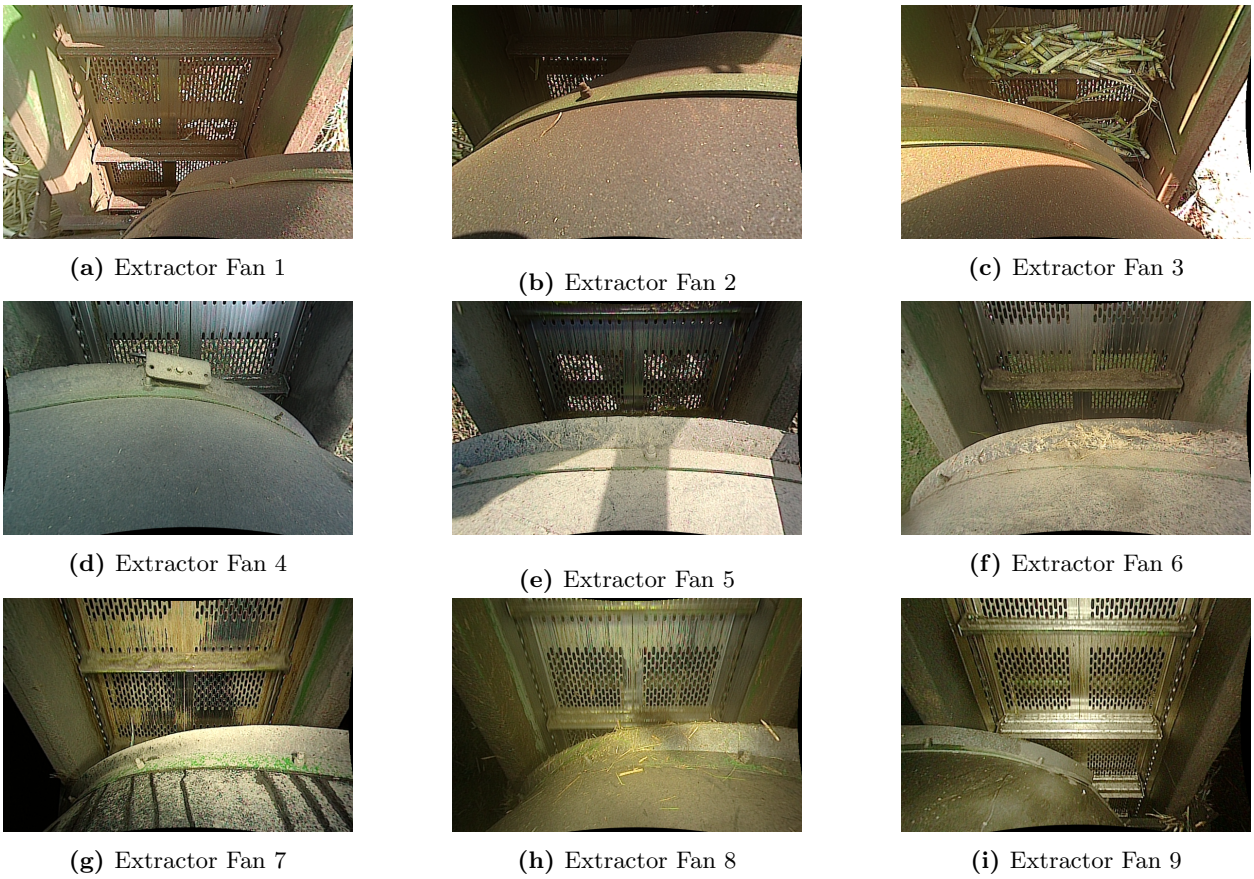
Figure 6.1: Data acquisition and pre-processing steps

6.1.3 Data complexity

Training a deep learning algorithm to estimate mass flow from video data is far more complex for the in-field harvesting application than a controlled laboratory scenario. Laboratory data or bamboo has a consistent yellowish color and the elevator background is always green, yet the only factor that affects bamboo images is lighting. Figure 6.2 shows different images of bamboo under different lighting conditions with some noticeable differences.

Figure 6.2: Bamboo images under different lighting conditions

Unlike laboratory experiments, field experiments were affected by more extreme ambient lighting variation, shadowing, dirt (on lens, material, background), elevator background color, material composition (e.g. green, burnt, root balls), billet and leaf size and color variation, larger changes in material density, extractor fan presence, airborne debris, and grossly overfilled slats. Figure 6.4 shows a set of images of the extractor fan blocking the view of the camera. It can be seen that images have different colors and lighting conditions.



(a) Extractor Fan 1 **(b)** Extractor Fan 2 **(c)** Extractor Fan 3

(d) Extractor Fan 4 **(e)** Extractor Fan 5 **(f)** Extractor Fan 6

(g) Extractor Fan 7 **(h)** Extractor Fan 8 **(i)** Extractor Fan 9

Figure 6.4: Different images of the extractor fan blocking the view of the camera

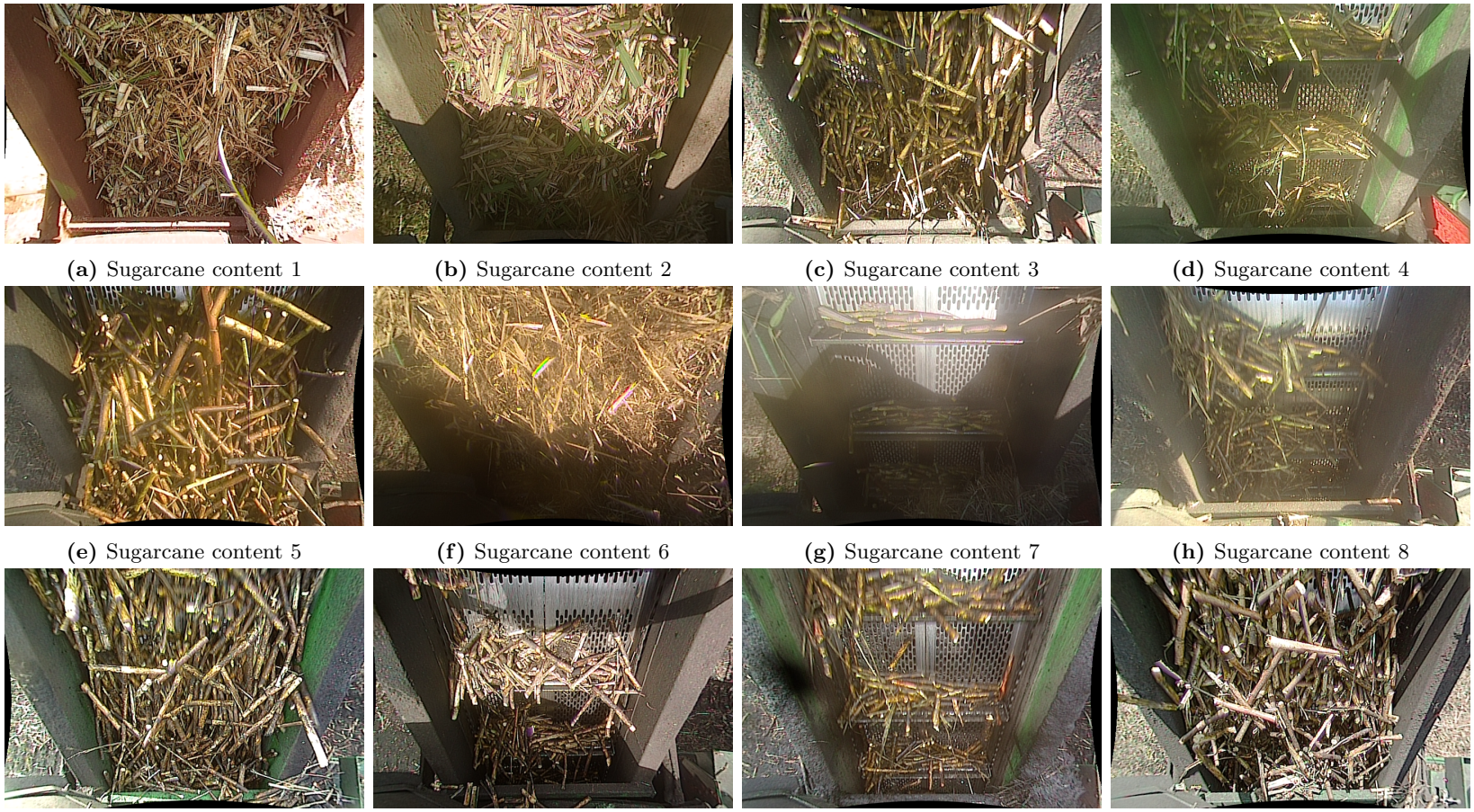
Even without the presence of extractor fan, the elevator running empty takes on many different colors due to dirt, rust, stripped paint, and exposure to varying lighting conditions as shown in Figure 6.6. Lastly, a set of images of different sugarcane content are shown in Figure 6.8. Observing these images it is shown how much more complex the sugarcane application compared to the proof of concept testing (bamboo) in Figure 6.2. Nonetheless, in this work it is demonstrated that the devised algorithm based on laboratory data can still learn to estimate mass despite all of these different complicating factors. Another factor to consider is the length of field logs/runs, which can be up to 26x larger than laboratory logs/runs. Thus, the deep neural network must learn to predict mass from very sparse ground truth, which affects the ability of convergence during training.

(a) Empty 1 **(b)** Empty 2 **(c)** Empty 3 **(d)** Empty 4

(e) Empty 5 **(f)** Empty 6 **(g)** Empty 7 **(h)** Empty 8

(i) Empty 9 **(j)** Empty 10 **(k)** Empty 11 **(l)** Empty 12

Figure 6.6: Empty elevator images with various colors and under different lighting conditions



(a) Sugarcane content 1 **(b)** Sugarcane content 2 **(c)** Sugarcane content 3 **(d)** Sugarcane content 4

(e) Sugarcane content 5 **(f)** Sugarcane content 6 **(g)** Sugarcane content 7 **(h)** Sugarcane content 8

(i) Sugarcane content 9 **(j)** Sugarcane content 10 **(k)** Sugarcane content 11 **(l)** Sugarcane content 12

Figure 6.8: Different sugarcane content (green and burnt) with various amounts shown to have different colors as exposed to sunlight

6.2 Learning mass from images

The methods presented in this section were previously presented in chapter 4, so we will briefly revisit them. You can skip to the next section if you have gone through these methods in chapter 4.

Learning mass from images is possible in this problem context due to the constraints imposed on the system and the ability to learn complex patterns via end-to-end training with deep learning. More specifically, the camera has a fixed sampling frequency that is fast enough to measure all the material passing by, it is mounted at a fixed distance from the elevator, and the only additional factor needed is slat velocity to scale the accumulated mass. Intuitively, the velocity is scaling the mass to produce a mass flow, since if the mass was sitting still it would not be accumulated. Alternatively it can be thought of as a way to account for frame overlap. Even though complex nonlinearities (presented in section 6.1.3) are present, they however can be learned by optimizing a deep neural network (DNN) due to fixed locations between the camera and the elevator.

The loss function shown in Equation 6.1 (presented previously in Equation 4.4) is also used as it is with field data to minimize the error between predicted and ground truth values. Another reason for sticking with MSE to model field data is that we considered mean absolute error (MAE) loss, but it did not perform as good as MSE.

$$L_i(x, y; w) = \frac{1}{n_i}\Big\{y_i - \sum_{j=1}^{n_i}(f(x_{ij}; w) \times v_{ij} \times t)\Big\}^2 \tag{6.1}$$

The same techniques presented in chapter 4 such as temporal smoothing (section 4.4) shown in Equation 6.2 and gradient update (section 4.5) are applied in same manner to field data.

$$L_i(x, y; w) = \frac{1}{n_i}\{y_i - \sum_{j=1}^{n_i}(f(x_{ij}; w) \times v_{ij} \times t)\}^2 + \frac{\lambda}{n_i}\sum_{j=1}^{n_i}\{f(x_{ij}; w) - f(x_{i(j-1)}; w)\}^2 \tag{6.2}$$

6.3 Model architecture and training procedure

RES9-ER architecture (section 4.6) which is the best performing architecture on laboratory data (shown in Figure 6.9) is also used to predict on field data. Thus far, with field data other options while training the model were explored.

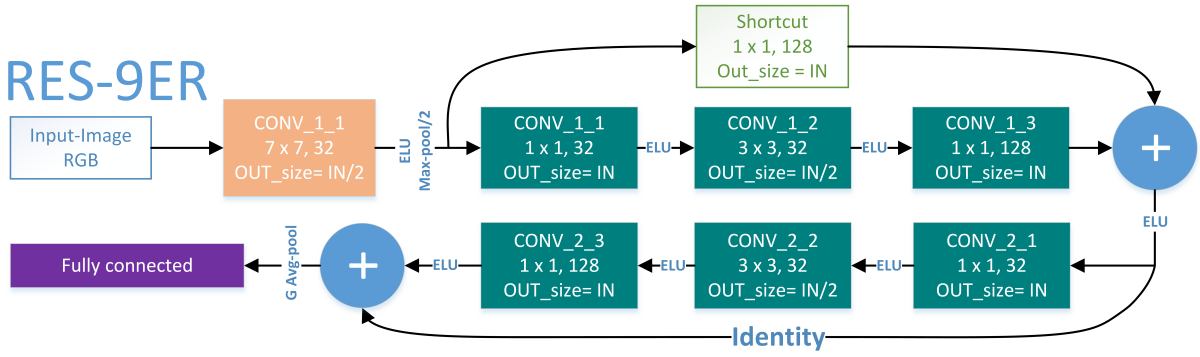


Figure 6.9: Reduced residual 9 architecture (RES-9ER)

Field data was split into training, validation, and testing sets for each individual region (Texas, Louisiana, Brazil, Florida, and all regions combined together) per Table 5.1. The split was done using scikit-learn package and the split ratios for train, validation, and test sets are 80,10,10 respectively. Since training using a deep residual neural network with images takes fairly a long time given the large dataset (over 3M images), transfer learning was utilized with the vision-based approach to cut the training times. To further optimize training speed, Tensorflow DATASET API was used to form the input pipeline and data was pre-processed and stored in the format of Tensorflow TFrecords on solid-state drives. This reduced the training time on the all-fields training set (obtained by combining fields from all the different regions) by a factor of ≈34x (57 days down to 1.7 days). The model was trained using Adam optimizer with a fixed learning rate of 0.001 and a batch size of 32.

Prior to training, field data was pre-processed by applying a mask on images to exclude irrelevant information and then the data was down-sampled (5^{th} of the original size) and normalized. Field data was initially trained using the "RES9-ER" model, but opportunities such as adding dropout layers were explored to see if it would help the model generalize better; however, adding drop-out did not seem to have any effect on the model performance and therefore was discarded. Furthermore, training on gray-scale images was considered, which did not obtain the level of accuracy as when the model was trained on colored images. Moreover, when attempted to train

on gray-scale images and the penalty term (temporal smoothness) was removed, that resulted in noisy signals as the example shown in Figure 6.10.

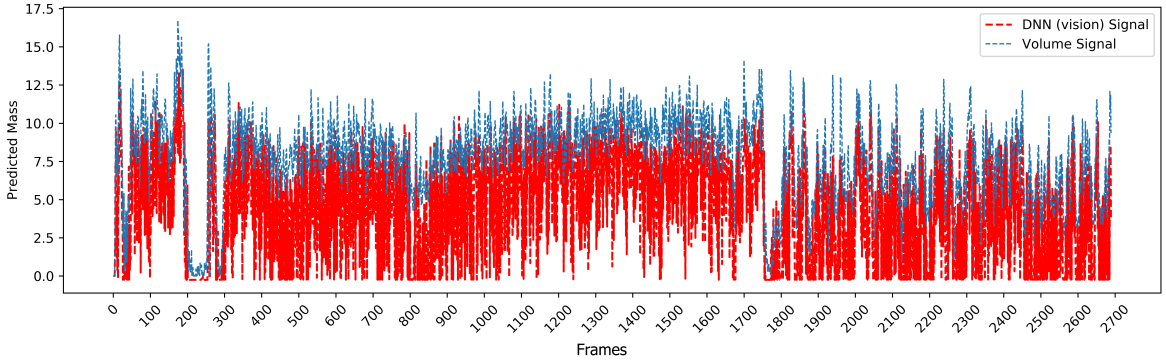


Figure 6.10: Overlay of vision estimated signal over volume estimated reference signal of a select run/log. The vision signal was predicted using gray-scale images and no temporal smoothing was applied to the signal

Training on RES-9E was also considered but it did not perform any better than RES-9ER. Further, training with deeper architectures required the use of batch-normalization layers [32], but the batch-normalization layers enforced a smoothing effect (shown in Figure 6.11) that affected the overall shape of the prediction signal, hence impacting the accuracy. In conclusion and after multiple training trials, the best performing architecture remains "RES-9ER" with only the penalty term added.

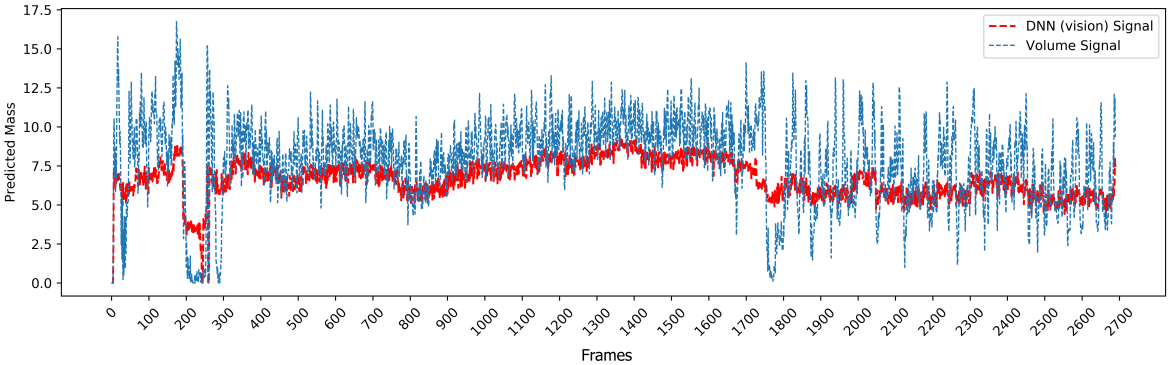


Figure 6.11: Overlay of vision estimated signal over volume estimated reference signal of a select run/log. The vision signal was predicted using a 16 layer architecture with batch normalization layers added

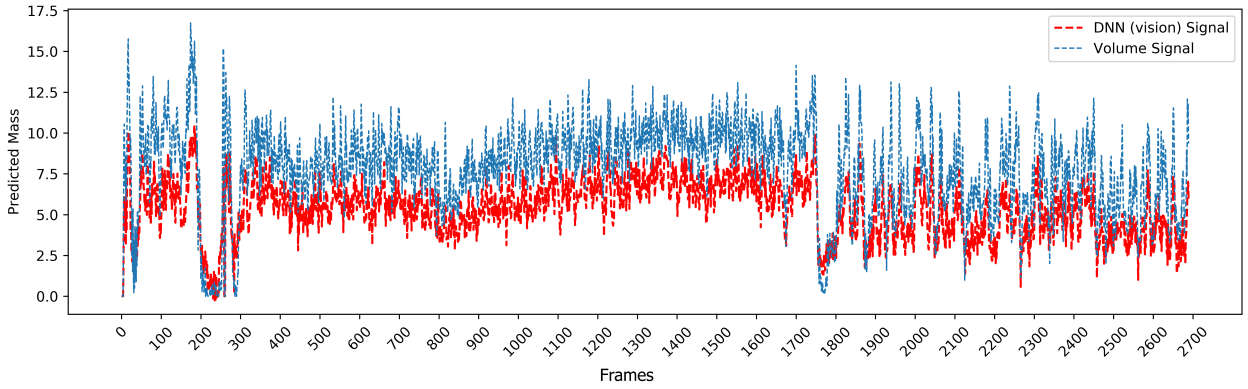


Figure 6.12: Overlay of vision estimated signal over volume estimated reference signal of a select run/log. The vision signal was predicted using RES-9ER and it shows a reasonable shape

6.4 Results and analysis

Field data (sugarcane) was evaluated using the same methods used with laboratory data (bamboo). The core training procedure for the field data followed the same procedure as the laboratory data highlighting that the proposed semi-supervised method is transferable. Field data that is evaluated using the volumetric-based approach was trained on a 4-layer feed-forward neural network to adjust for nonlinear changes in material density that correlate with volume. Volumetric-based estimates serve as a baseline comparison to the vision-based method.

6.4.1 Predicted signals investigation

Scoring accurate predictions in terms of how close the accumulated mass compared to the ground truth measurement of each run/log is feasible and interpretable; however, it does not guarantee correct predictions of mass flow per image since the chance exists that the model could learn non-generalizable patterns. To investigate that the model was learning the proper features and going to generalize, the output signal from the DNN was compared against the volume estimation signal from the stereo camera per run/video. It is emphasized that, while the volume signal may not accurately reflect mass without an accurate density estimate, the relative shape of the volume

signal is going to correlate with mass flow strongly. Thus, the volume signal can be used to check that the mass flow estimate from images is reasonable.

Figure 6.13 shows an example run where the DNN misidentifies empty spots and other regions (circled in green). This run is from Florida dataset which initially lacked empty or zero runs in it, so the network could not learn such behavior. To remedy this problem we sourced new empty runs and retrained the DNN to give a better overall prediction as shown in Figure 6.14.

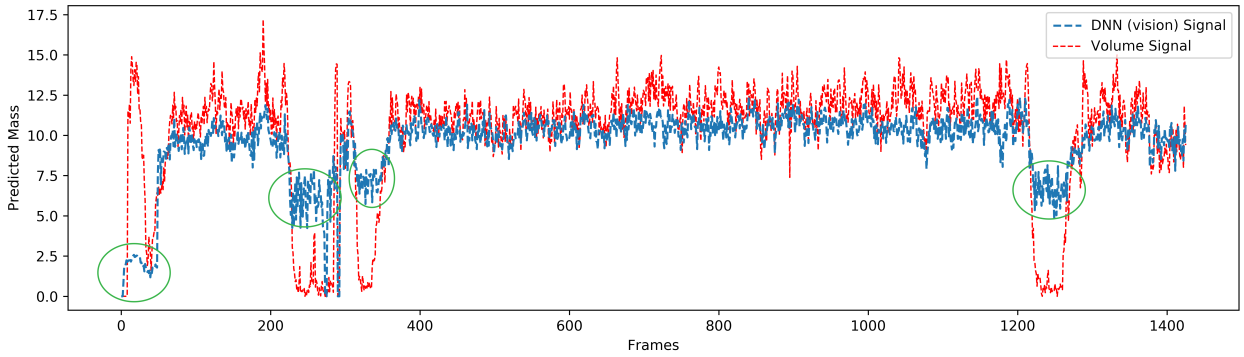


Figure 6.13: Training without zero runs resulted in poor prediction of approximate signal shape

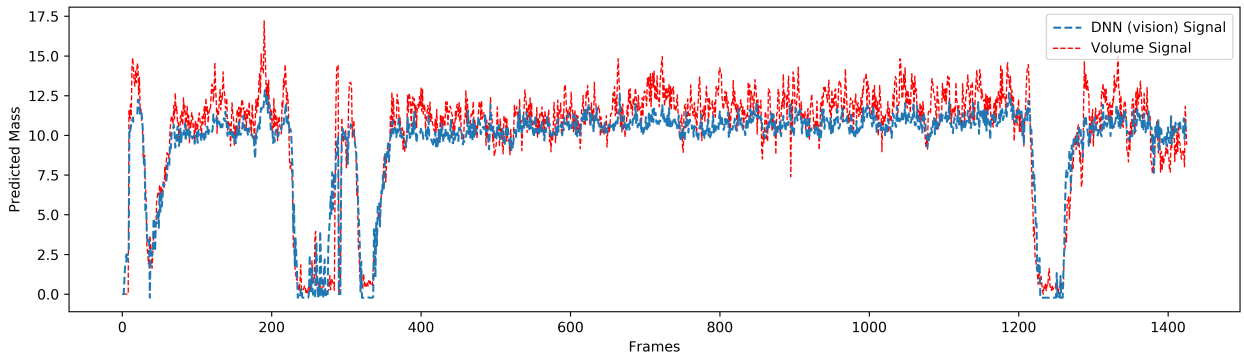


Figure 6.14: Better prediction of signal shape is maintained with the use of empty runs

Another instance that was observed to affect the signal shape is when the extractor fan blocks part of the camera view as shown in Figure 6.15. This situation occurs when the elevator is running at very low speed or stationary with usually no flowing material.

The presence of the extractor fan makes the DNN add noise to the predicted signal (area circled

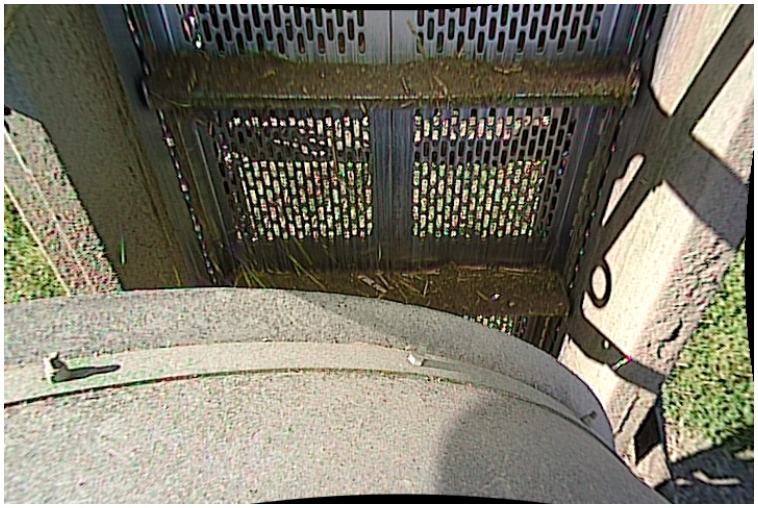


Figure 6.15: Image of the extractor fan from an example run/log that is affected from the presence of the extractor fan

in red in Figure 6.16) at empty spots, where it is supposed to be straight flat. This is not an extreme situation and does not significantly impact the accuracy of the system overall, and can be easily remedied by referencing a signal of the extractor fan and elevator positions via CANbus.

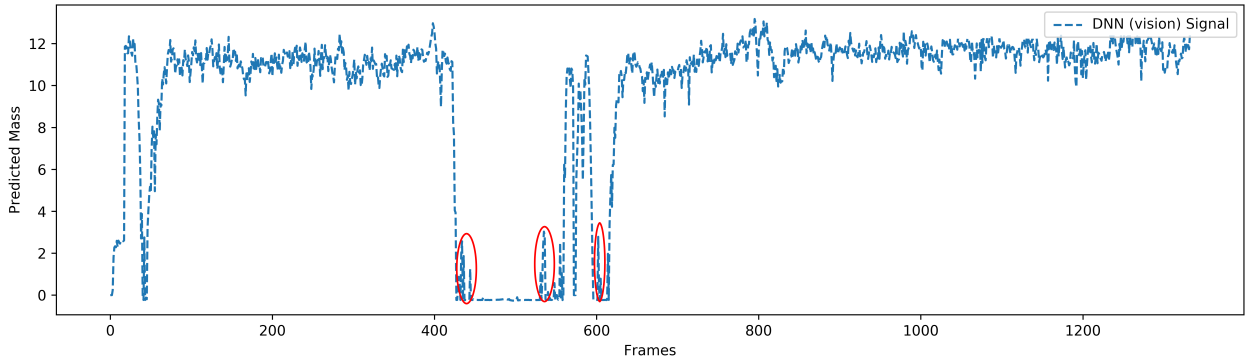


Figure 6.16: Noise observed at empty spots in the predicted signal as the extractor fan blocks part of the camera view

To further demonstrate that material flow is being captured correctly, Figures { 6.17, 6.18, 6.19, and 6.20 } show overlays of the predicted vision signal on top of the reference volume signal for a select run from each region. The selected runs show intermittent, incremental/decremental, and

very lengthy material flows. In all scenarios, it is shown that the vision signal is smoother than the volumetric signal and that is due to applying temporal smoothing in the learning process as well as the learned features from images are consistent especially when no material is present. It is seen in Figure 6.17 that the DNN successfully captures the sudden presence of material (spikes). Figure 6.18 shows an example of consistent material flow.

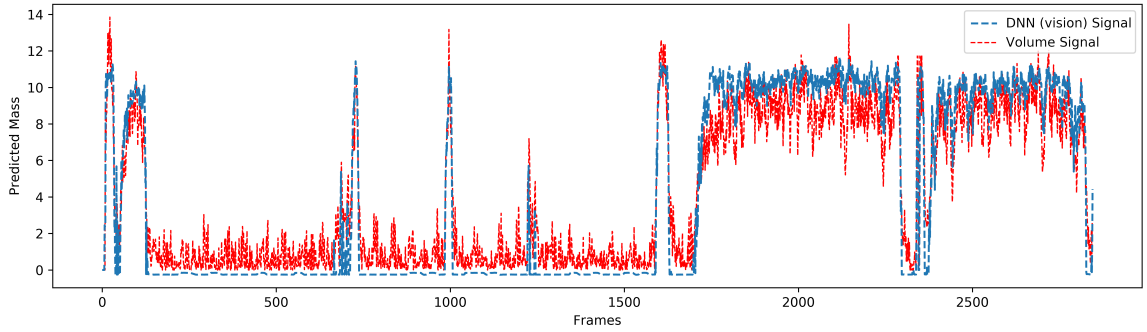


Figure 6.17: Selected run from Louisiana dataset showing intermittent spiky material flow

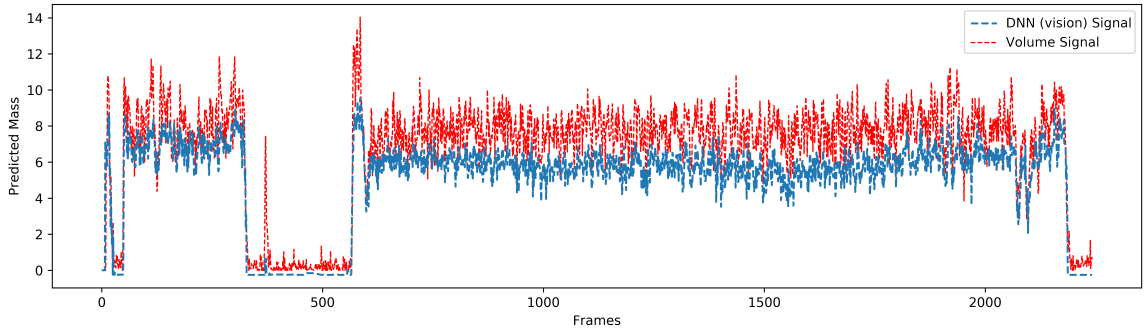


Figure 6.18: Selected run from Texas dataset showing consistent material flow

Figure 6.19 shows that the DNN signal successfully captures the fine details of the signal shape as material flow changes with time (increase and decrease). The vision and volumetric signals correlate a lot with each other except that the vision signal is smoother and that is again due to the temporal smoothing effect.

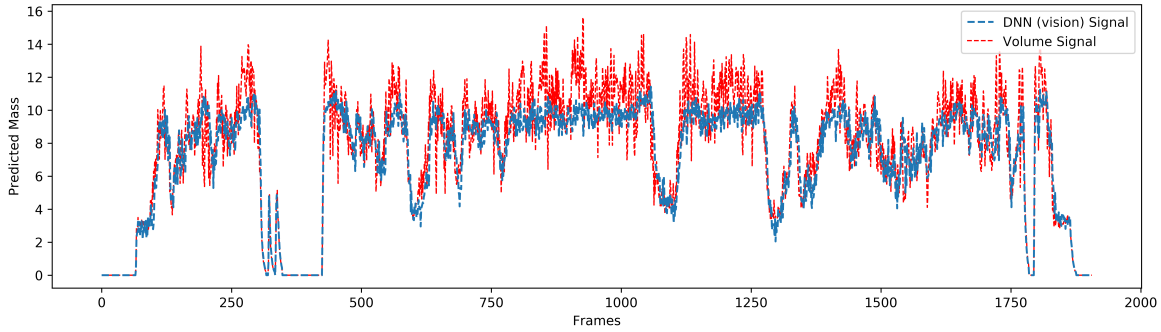


Figure 6.19: Selected run from Brazil dataset showing incremental and decremental material flow

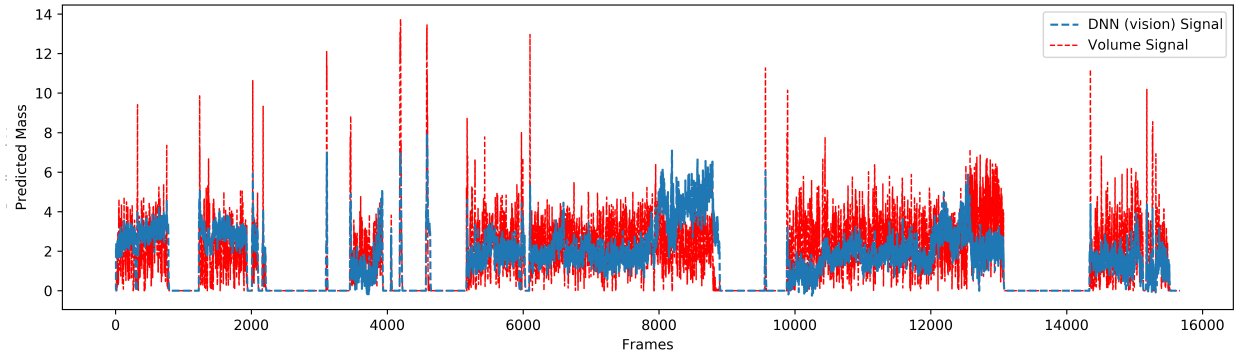


Figure 6.20: Selected run from Florida dataset showing very lengthy intermittent material flow

6.4.2 Overall performance

Field data is evaluated on each region independently as well as on all fields from the regions combined. The performance is reported for the test-set of each region. Table 6.2 summarizes the average error of both the volumetric (the neural network method, which was shown to outperform the box-cox transformation method) and the vision-based approaches for the different regions independently and for all the data as whole. Note that in order to convert volume to mass, a calibration coefficient (density) is estimated implicitly via a feed-forward neural network. Some extreme outliers were found in the case of Florida and All-fields data and are included in reported average error. From Table 6.2 it can be seen that the vision approach outperforms the volumetric approach in each of the regions and when all regions are combined. Florida seemed to have high average error compared to other regions because it included both burnt and green cane

Table 6.2: Test-set average error per region.

	Test-set average error	
Region	**Vision**	**Volumetric**
Louisiana	6.22%	10.54%
Texas	**5.88%**	7.78%
Brazil	8.76%	8.86%
Florida	14.65%	15.78%
All fields	15.18%	25.12%

as well as due to the presence of high trash content. The volumetric approach fared poorly when evaluated on all-fields and that is due to the nature of volumetric signal being it highly dependent on the estimated density which varies by material type (green or burnt). The vision approach performed quite well on Louisiana data and that is due to not relying on density to estimate mass unlike the case with volumetric-based estimation. Further, variations between regions exist due to the different machine operation and environmental conditions.

Given the different environmental factors and machine operation settings, the vision approach offered better average error when combining all regions together (All-fields) with $\approx 10\%$ improvement compared to the volumetric overall average error. This demonstrates the robustness of the vision based method when posed with different types of material and environmental conditions.

6.4.3 Outliers investigation

In section 5.1.3 it was noted that the point cloud estimation (volumetric estimation) was affected by rare instances when material flows were very high and the trash, especially dry and light, could fly up and block the camera view as well as when the sun would shine directly into the camera and wash out portion of the point cloud. Since the stereo camera was also used to produce the images used in the vision analysis, the vision system also was impacted by these instances.

Generally, the vision system estimates were biased low for some runs that had large sunshine blob present in images, and biased high when there were high trash content in images. The latter situation can be improved in future studies where another algorithm can work concurrently to estimate the amount of trash present and then use such information in the DNN mass estimation process.

Figure 6.21 shows overlay of error distributions of both vision and volume estimates of the all-fields test set. The outliers (circled in red) in the distribution were due to bad speed sensor measurements.

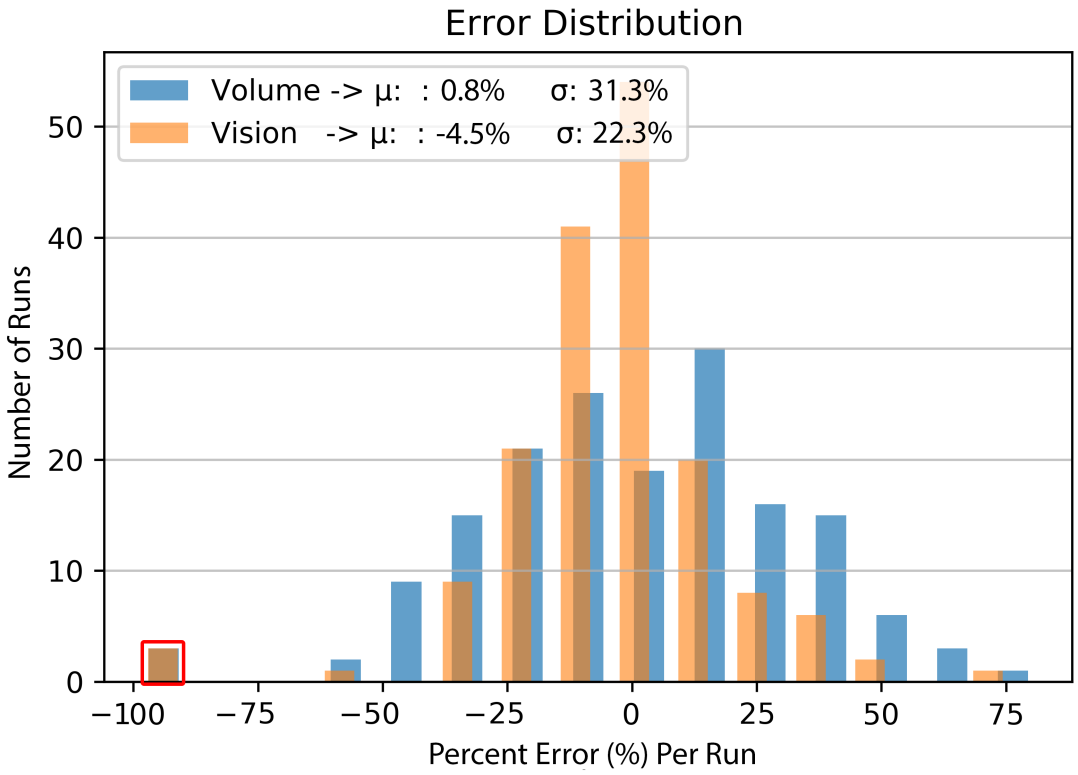


Figure 6.21: Histogram distribution of error of volume and vision estimates on all-regions test set

The matching algorithm that was used to estimate volume from point cloud resulted in saturated volume estimates at high material flows. To investigate whether this phenomena would also occur

with images, we plot the error against mass flow for the all-fields test set. Plotting error against mass flow also gives insights into potential outliers as well as under/over estimates. As shown in Figure 6.22 no trend is observed between error and volume flow, which demonstrates that the vision solution is not impacted by the high material flows unlike the volumetric solution.

From Figure 6.22 we have identified some runs to have bad speed sensor measurement, one run with a bad ground truth measurement, one run is under-estimated due to being exposed to direct sunlight, and some runs are over-estimated due to the presence of high trash content. It is again emphasized that the cases of high trash content can likely be remedied with the inclusion of trash prediction algorithm in the training process.

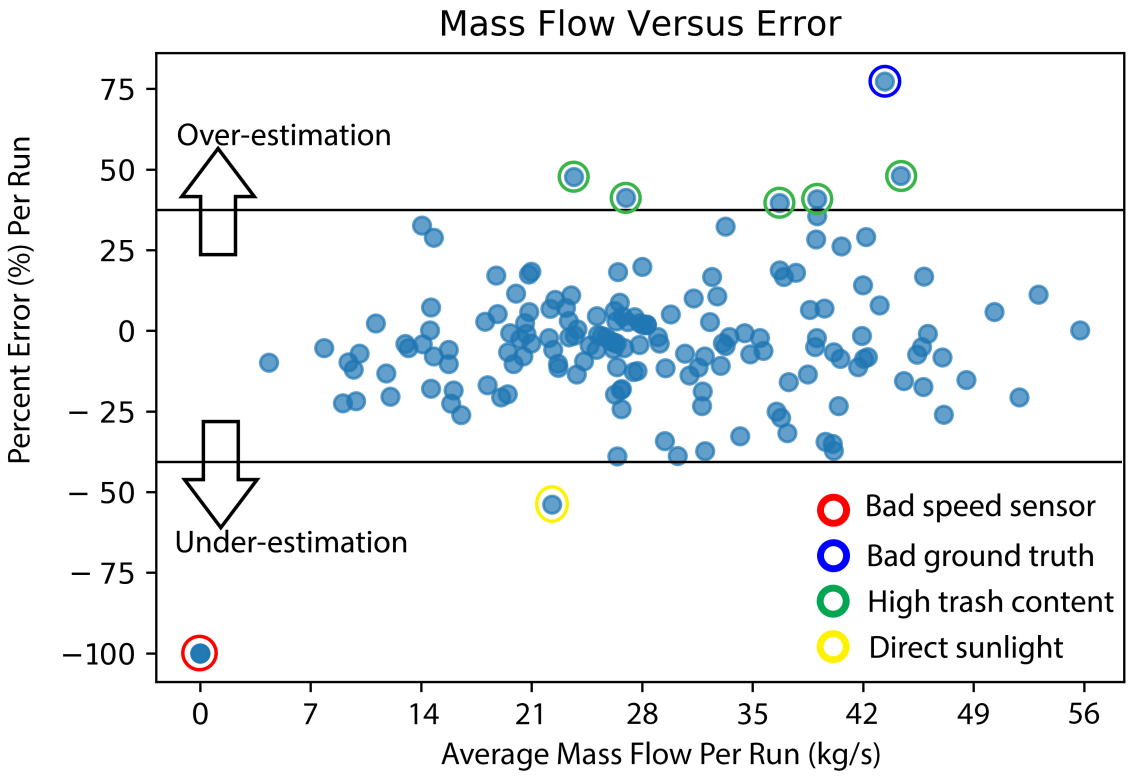


Figure 6.22: Mass flow vs. error for all-fields test set

Results summary

- ✓ Developed a generalizable deep learning semi-supervised learning algorithm that was tested on two different materials (bamboo and sugarcane).
- ✓ The developed algorithm outperforms older methods (volumetric estimation) with a remarkable difference of over 10% improvement when tested on sugarcane data of different regions, seasons, and different material composition.
- ✓ The algorithm scored an average error as low as 5.88% on a select season.
- ✓ Transfer learning was successfully applied and yielded in faster training times.
- ✓ Rigorously evaluated on real life data ($\approx$ 7000 Mg)

6.5 Conclusion

In this work, we presented a generalizable semi-supervised leaning algorithm that makes inference on mass flow of material from sequences of images by training a deep neural network with very sparse ground truth. The proposed method showed improvements over older and more expensive methods that must first acquire a volumetric estimate of the material and then calibrate the density. The presented algorithm was tested on two different materials and under controlled and real operation environments. The results obtained herein demonstrate that the algorithm is readily transferable to other crops or even applications. Transfer learning was successfully applied and helped reduce the time required to train the deep learning model. Lastly, prediction on our specific application (sugarcane) can be further improved by applying unsupervised learning methods. It is possible to train disentangled representations of features such as trash content, which would remedy the issue of over-estimation due to the presence of high trash content.

CHAPTER 7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

In this work we attempted to answer the following research questions: (1) What are the deep learning techniques that can be utilized to capture a physical property (mass) of flowing material through only processing sparsely annotated images? (2) To what extend can transfer learning be applied from one application or material to another, and (3) How much of change is required and/or what requirements should be maintained to achieve generalization?. To address each question properly we followed a systematic approach where our work has gone several phases of implementation as shown in Figure 7.1.

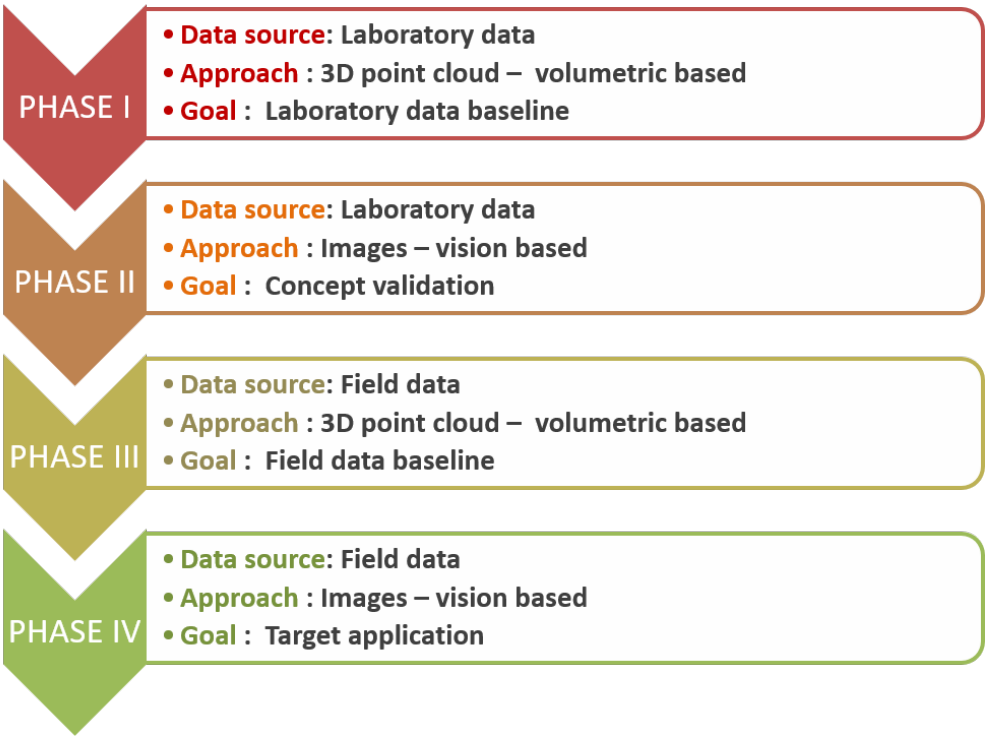


Figure 7.1: Summary of project phases

In phase I, we followed a volumetric-based approach to estimate mass flow through generating a 3d point cloud volumetric representation of flowing material. In this approach, bamboo was used as a surrogate material to sugarcane under controlled environment setup and was maintained as a baseline comparison to a vision-based approach via deep learning. Mass was estimated using the volume measurements produced by the 3d point cloud from the stereo camera along side a calibration factor (density) when applying monotonic transformation to the volumetric data. However, mass was directly estimated when using a feed-forward neural network as the neural network estimates density implicitly. The system achieved an R^2 of 97.4% when fitting average volume flow per test against average mass flow after correcting for bulk density changes with volume. When neglecting runs with low light (<2.7 lux) the system achieved an average error of -4.8% with a standard deviation of 11% when using a neural network. Results reported in this phase were used as a baseline comparison to the vision method under controlled environment setup in phase II.

In phase II, we developed and implemented a novel vision-based approach to estimate mass flow through a deep neural network that quantifies mass from images. This approach was conducted on bamboo and maintained under controlled environment setup to initially assess the feasibility of the vision-based method. Further, in this phase we compared against results obtained in phase I to support the development of a vision-based approach for a sugarcane yield monitor on sugarcane harvesters. A semi-supervised algorithm that makes inference on mass flow of material from sequences of images by training a deep neural network with sparse ground truth showed improvements over older and more expensive methods that must first acquire a volumetric estimate of the material and then calibrate the density. With the careful DNN design, an average error of 4.5%, better signal shape, and high frame rate up to 348FPS were obtained. Additionally, a dynamic algorithm was devised to accommodate the memory limitations of GPU working with large sequences of images as well as a temporal smooth technique was devised to improve the accuracy of predicted signals. Moreover, a visualization technique was introduced to investigate what the DNN algorithm learns behind the scenes. The results obtained in phase II supported

applying the methods derived to predict mass flow of field harvested sugarcane as measured on machine [phase IV], which experiences many other complicating factors affecting density and visual appearance of the material.

In phase III, a volumetric-based approach was followed to estimate mass flow through generating a 3d point cloud volumetric representation of flowing material. This approach was conducted on sugarcane under real environment operation and was maintained as a baseline comparison to a vision-based approach via deep learning. The concept is contingent on a calibrated value, or density, to convert from volume to mass when evaluated using box-cox transformation. The calibration values are relatively easy to obtain by tracking volume into a semi and getting a weight back from the mill. However, mass was directly estimated when evaluated using a feed-forward neural network. When separated out by season, region, and crop type the system performance is acceptable, achieving an average error of 7.88% when mass is estimated via neural network. Given the large density changes observed with varying trash levels, use of point clouds to quantify and adjust for trash could have a very large impact on system performance. Additionally, the volumetric approach seemed to perform better when data is modeled based on region; however, this implies multiple calibration values. Finally, this work was used as a baseline comparison for solving the same problem using visual processing (images) via a deep neural network (phase IV).

In phase IV, transfer learning was applied to the task studied in phase II and then we assessed the required changes such as DNN architecture redesign or techniques embedded in the training procedure like adding batch normalization or dropout layers. This approach was conducted on sugarcane and maintained under real environment operation. Throughout the thorough investigation of system performance in this phase, the developed algorithm showed superior performance over older methods (volumetric estimation) with a remarkable difference of over 10% improvement when tested on sugarcane data from different regions, season, and of different material composition. The algorithm achieved an average error of 5.88% on a select season and transfer learning was successfully applied and yielded faster training times given the large dataset of ours. The presented algorithm was tested on two different materials and under controlled and

real operation environments. The results obtained demonstrated that the algorithm is readily transferable to other crops or even applications. Figure 7.2 illustrates the dependencies between all the phases, where boxes in orange represent intermediate results and Green boxes represent final results which are the outcomes of this research.

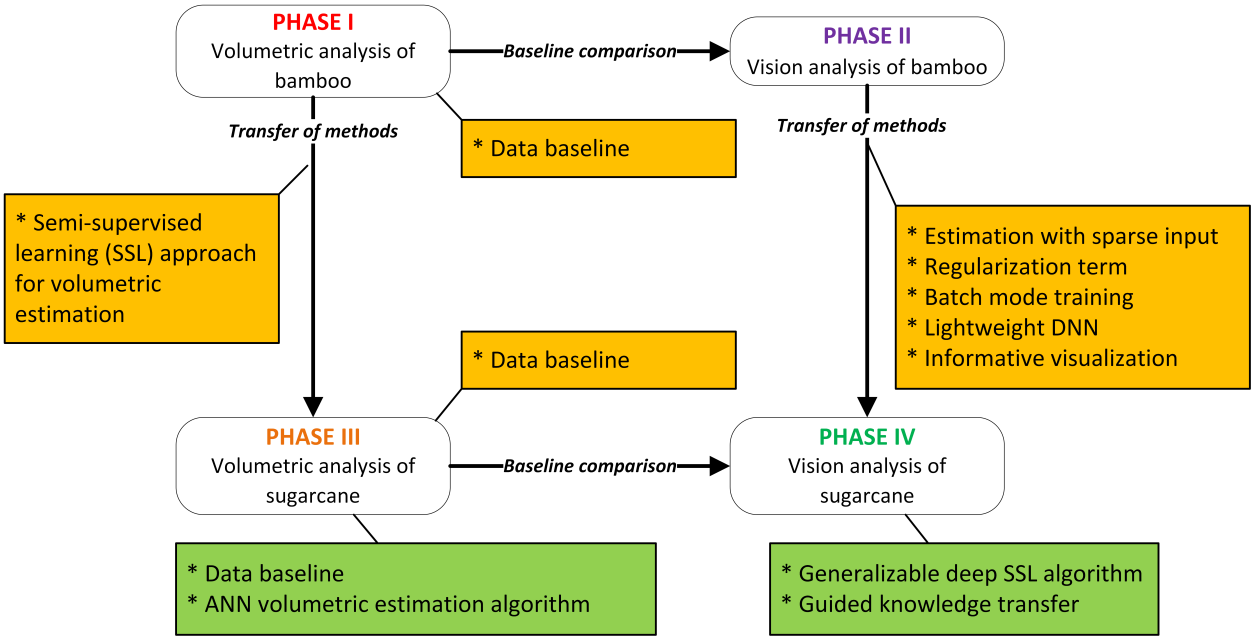


Figure 7.2: Phases dependency map

To ensure a successful transfer of knowledge, a list of guidelines was set. Following these guidelines shall support the transfer of the methods presented in this work to other crops or perhaps applications. These guidelines were derived from our extensive experimentation and research as we progressed in solving our problem. These guidelines are summarized as follows:

- Sufficient number of empty runs must be maintained as part of the design of experiment. This is critical factor to identifying empty spots in runs.
- Camera should be placed at fixed suitable location from material with a good visual of the target material properties.

- It is preferred not to have lengthy runs in the design of experiment. Shorter runs mean less sparse ground truth which helps the optimization process learn faster. However, this mean more runs, which implies more ground truth measurements. In our work, run lengths varied between 1500 to 20000 frames, so something in the middle should be suitable.

- It is preferred to have consistent run lengths as this would help the optimization process learn faster. What matters is obtaining accurate prediction on the data point level not on the run level.

- We recommend 60/20/20 - train/test/validation data split for small data sets (like our laboratory data) and 80/10/10 for large data sets (like our field data).

- If designing a new DNN architecture, including batch normalization would help the network converge faster especially if it is deep but it might affect the shape of the signal, hence the accuracy.

- If designing a new DNN architecture, adding dropout layers might help prevent over-fitting but may not improve accuracy.

- Training with gray-scale images is possible and can score relatively good compared to colored images.

7.2 Future work

A possible future direction of our work could be investigating unsupervised learning techniques to characterize material in images, e.g. identifying the amount of trash to material ratio (e.g sugarcane) present in an image without using any labeled data. A variational autoencouder is likely to be trained to disentangle representations of features in images such as trash content. Identifying trash content can greatly improve the prediction accuracy and remedy the issue of over-estimation as a signal can be routed back to the primary DNN estimator to account for trash presence in images, thus making more accurate estimates. Learning trash content can also help in

feedback control of the primary extractor fan where this can lead to optimizing the cleaning process.

Besides using an autoencoder, learned features can be investigated by tracking the outputs prior the final layer in the deep neural network. Variability plots can be used to draw conclusions on how the nodes (neurons) in the target layer relate to each other. Further, we could investigate what specific information do the nodes represent as ultimately combining the nodes output together aim to estimate mass, therefore, there could be a possibility to characterize what is being detected in the image.

Another possible opportunity is to characterize density for volume estimates in a similar manner. Learning such information at no cost of labeling data can aid the prediction process through making more confident inferences as supported by more useful information.

We also wish to transfer the method presented herein to other crops to see how the methods generalize with crops that are completely different than sugarcane and bamboo in shape and texture.

Lastly, we could perform structured pruning on the DNN model to further reduce the number of usable parameters and devise an efficient implementation on custom hardware, such as an FPGA, that takes advantage of model sparsity, which ultimately improve execution time.

BIBLIOGRAPHY

[1] Technology Ag Leader. Grain yield monitor, 1992.

[2] S Al-Thyabat, NJ Miles, and TS Koh. Estimation of the size distribution of particles moving on a conveyor belt. *Minerals Engineering*, 20(1):72–83, 2007.

[3] Rie Kubota Ando and Tong Zhang. A high-performance semi-supervised learning method for text chunking. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 1–9. Association for Computational Linguistics, 2005.

[4] Maria-Florina Balcan, Avrim Blum, Patrick Pakyan Choi, John D Lafferty, Brian Pantano, Mugizi Robert Rwebangira, and Xiaojin Zhu. Person identification in webcam images: An application of semi-supervised learning. 2009.

[5] Horace B Barlow. Unsupervised learning. *Neural computation*, 1(3):295–311, 1989.

[6] Kristin P Bennett and Ayhan Demiriz. Semi-supervised support vector machines. In *Advances in Neural Information processing systems*, pages 368–374, 1999.

[7] Domingos GP Cerri and Paulo Graziano Magalhães. Sugar cane yield monitor. In *2005 ASAE Annual Meeting*, page 1. American Society of Agricultural and Biological Engineers, 2005.

[8] Pramod Chandrayan. Deep learning: Deep belief network fundamentals. https://codeburst.io/deep-learning-deep-belief-network-fundamentals-d0dcfd80d7d4, 2018. Accessed: 2019-11-09.

[9] Chen Change Loy, Shaogang Gong, and Tao Xiang. From semi-supervised to transfer counting of crowds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2256–2263, 2013.

[10] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

[11] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader, and Vineeth N Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847. IEEE, 2018.

[12] Anna Chlingaryan, Salah Sukkarieh, and Brett Whelan. Machine learning approaches for crop yield prediction and nitrogen status estimation in precision agriculture: A review. *Computers and Electronics in Agriculture*, 151:61–69, 2018.

[13] Peter Christiansen, Lars Nielsen, Kim Steen, Rasmus Jørgensen, and Henrik Karstoft. Deepanomaly: Combining background subtraction and deep learning for detecting obstacles and anomalies in an agricultural field. *Sensors*, 16(11):1904, 2016.

[14] G Cox, H Harris, and R Pax. Development and testing of a prototype yield mapping system. In *Proceedings-Australian Society of Sugar Cane Technologists*, pages 38–43. WATSON FERGUSON AND COMPANY, 1997.

[15] Graeme Cox, H Harris, R Pax, and R Dick. Monitoring cane yield by measuring mass flow rate through the harvester. In *Proceedings-Australian Society of Sugarcane Technologies*, pages 152–157. Watson Ferguson and Company, 1996.

[16] Graeme J Cox. *A yield mapping system for sugar cane chopper harvesters*. PhD thesis, University of Southern Queensland, 2002.

[17] Fabio G Cozman, Ira Cohen, and Marcelo C Cirelo. Semi-supervised learning of mixture models. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 99–106, 2003.

[18] James Cryer and the Huddle development team. Image analysis and comparison. https://github.com/rsmbl/Resemble.js, 2018. Accessed: 2019-04-28.

[19] Clement Douarre, Richard Schielein, Carole Frindel, Stefan Gerth, and David Rousseau. Deep learning based root-soil segmentation from x-ray tomography. *bioRxiv*, page 071662, 2016.

[20] Richard Earl, Paul N Wheeler, B Simon Blackmore, and Richard J Godwin. Precision farming–the management of variability. *Landwards*, 1996.

[21] Jeffrey Erman, Anirban Mahanti, Martin Arlitt, Ira Cohen, and Carey Williamson. Offline/realtime traffic classification using semi-supervised learning. *Performance Evaluation*, 64(9-12):1194–1213, 2007.

[22] GTi Gasser. The volume rate measuring system, 9 2012.

[23] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[24] Danping Guo. Kernel and bulk density changes due to moisture content, mechanical damage, and insect damage. 2015.

[25] Olivier Guyot, Thierry Monredon, David LaRosa, and Alain Broussaud. Visiorock, an integrated vision technology for advanced control of comminution circuits. *Minerals Engineering*, 17(11-12):1227–1235, 2004.

[26] Muhammad Hamdan. Mass estimation implementation. https://github.com/mhamdan91/Mass_Flow_Estimation, 2019. Accessed: 2019-06-07.

[27] Muhammad Hamdan, Diane Rover, Matthew Darr, and John Just. Mass estimation from images using deep neural network and sparse ground truth. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pages 1987–1992. IEEE, 2019.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

[29] Geoffrey E Hinton, Terrence Joseph Sejnowski, and Tomaso A Poggio. *Unsupervised learning: foundations of neural computation*. MIT press, 1999.

[30] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[31] Erik Hulthén. *Real-time optimization of cone crushers*. Chalmers University of Technology, 2010.

[32] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[33] U Jadhav, LR Khot, R Ehsani, V Jagdale, and JK Schueller. Volumetric mass flow sensor for citrus mechanical harvesting machines. *Computers and electronics in agriculture*, 101:93–101, 2014.

[34] Shruti Jadon. Introduction to different activation functions for deep learning. https://medium.com/@shrutijadon10104776/survey-on-activation-functions-for-deep-learning-9689331ba092, 2018. Accessed: 2019-11-09.

[35] Marcus Häfner Karl-Hans Kromer, Peter Degen and Bonn Oliver Schmittmann. Spatially-specific yield measurement with sugar beet, 1 6 2001. 56 Landtechnik 1/2001.

[36] Kathrin. Recurrent neural networks (rnns) and long short term memory networks (lstms). https://www.knime.com/blog/text-generation-with-lstm, 2018. Accessed: 2019-11-09.

[37] Nataliia Kussul, Mykola Lavreniuk, Sergii Skakun, and Andrii Shelestov. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 14(5):778–782, 2017.

[38] Yevhen Kuznietsov, Jorg Stuckler, and Bastian Leibe. Semi-supervised deep learning for monocular depth map prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6647–6655, 2017.

[39] Erik G Learned-Miller. Introduction to supervised learning. *I: Department of Computer Science, University of Massachusetts*, 2014.

[40] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[41] Andy Lee. Comparing deep neural networks and traditional vision algorithms in mobile robotics. *Swarthmore University*, 2015.

[42] Sue Han Lee, Chee Seng Chan, Paul Wilkin, and Paolo Remagnino. Deep-plant: Plant identification with convolutional neural networks. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 452–456. IEEE, 2015.

[43] Yu Xuan Lee. Converting a simple deep learning model from pytorch to tensorflow. https://towardsdatascience.com/converting-a-simple-deep-learning-model-from-pytorch-to-tensorflow-b6b353351f5d, 2018. Accessed: 2019-11-09.

[44] Guangxia Li, Kuiyu Chang, and Steven CH Hoi. Multiview semi-supervised learning with consensus. *IEEE Transactions on Knowledge and Data Engineering*, 24(11):2040–2051, 2011.

[45] Pierre Lison. An introduction to machine learning. *Language Technology Group (LTG), 1*, 35, 2015.

[46] Lufeng Liu, Ye Yuan, Wei Deng, and Shuixiang Li. Evolutions of packing properties of perfect cylinders under densification and crystallization. *The Journal of chemical physics*, 149(10):104–503, 2018.

[47] Venkatesan M. Artificial intelligence vs. machine learning vs. deep learning. https://www.datasciencecentral.com/profiles/blogs/artificial-intelligence-vs-machine-learning-vs-deep-learning, 2018. Accessed: 2019-11-09.

[48] PSG Magalhães and DGP Cerri. Yield monitoring of sugar cane. *Biosystems Engineering*, 96(1):1–6, 2007.

[49] J.R. Magness, G.M. Markle, and C.C. Compton. *Food and Feed Crops of the United States: A Descriptive List Classified According to Potentials for Pesticide Residues*. Number nos. 828-838 in Bulletin (New Jersey Agricultural Experiment Station). New Jeresy Agricultural Experiment Station, 1971.

[50] Mike Mailander, Caryn Benjamin, Randy Price, and Steven Hall. Sugar cane yield monitoring system. *Applied engineering in agriculture*, 26(6):965–969, 2010.

[51] SK Mathanker, JC Buss, H Gan, JF Larsen, and Alan Christopher Hansen. Stem bending force and hydraulic system pressure sensing for predicting napiergrass yield during harvesting. *Computers and Electronics in Agriculture*, 111:174–178, 2015.

[52] Ishan Misra, Abhinav Shrivastava, and Martial Hebert. Watch and learn: Semi-supervised learning for object detectors from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3593–3602, 2015.

[53] JP Molin and LAA Menegatti. Field-testing of a sugar cane yield monitor in brazil. In *2004 ASAE Annual Meeting*, page 1. American Society of Agricultural and Biological Engineers, 2004.

[54] Allen Myers. Method and apparatus for measuring grain mass flow rate in harvesters, 1994. US Patent 5343761A.

[55] Petteri Nevavuori, Nathaniel Narra, and Tarmo Lipping. Crop yield prediction with deep convolutional neural networks. *Computers and Electronics in Agriculture*, 163:104859, 2019.

[56] NB Pagnano and PG Magalhaes. Sugarcane yield measurement. faculdade de engenharia agricola unicamp campinas sp, brazil 13083-970. In *Proceeding of 3rd European Conference on Precision Agriculture*, pages 839–844, 2001.

[57] Peng. Fully-connected, locally-connected and shared weights layer in neural networks easy explained. https://pennlio.wordpress.com/2014/04/11/fully-connected-locally-connected-and-shared-weights-layer-in-neural-networks/, 2019. Accessed: 2019-11-09.

[58] Maria Pérez-Ortiz, JM Peña, Pedro Antonio Gutiérrez, Jorge Torres-Sánchez, César Hervás-Martínez, and Francisca López-Granados. A semi-supervised system for weed mapping in sunflower crops using unmanned aerial vehicles and a crop row detection method. *Applied Soft Computing*, 37:533–544, 2015.

[59] MA Pierossi and SJ Hassuani. Caçamba instrumentada para pesagem de cana picada. *Proceeding of Semin Rio Copersucar De Tecnologia Agronomica*, 7, 1997.

[60] Randy R Price, Richard M Johnson, and Ryan P Viator. An overhead optical yield monitor for a sugarcane harvester based on two optical distance sensors mounted above the loading elevator. *Applied engineering in agriculture*, 33(5):687–693, 2017.

[61] Randy R Price, John Larsen, and Alex Peters. Development of an optical yield monitor for sugar cane harvesting. In *2007 ASAE Annual Meeting*, page 1. American Society of Agricultural and Biological Engineers, 2007.

[62] RR Price, RM Johnson, RP Viator, J Larsen, and A Peters. Fiber optic yield monitor for a sugarcane harvester. *Transactions of the ASABE*, 54(1):31–39, 2011.

[63] Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. *WACV/MOTION*, 2, 2005.

[64] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[65] Sumit Saha. A comprehensive guide to convolutional neural networks. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53, 2018. Accessed: 2019-11-09.

[66] Lalit Saxena and Leisa Armstrong. A survey of image processing techniques for agriculture. 2014.

[67] JUNSU Shin. *Mass and Size Estimation of citrus Fruit by Machine vision and citrus greening diseased fruit detection using spectral analysis*. PhD thesis, University of Florida, 2012.

[68] Claudia Brito Silva, Márcia Azanha Ferraz Dias de Moraes, and José Paulo Molin. Adoption and use of precision agriculture technologies in the sugarcane industry of são paulo state, brazil. *Precision agriculture*, 12(1):67–81, 2011.

[69] Arti Singh, Baskar Ganapathysubramanian, Asheesh Kumar Singh, and Soumik Sarkar. Machine learning for high-throughput stress phenotyping in plants. *Trends in plant science*, 21(2):110–124, 2016.

[70] Srdjan Sladojevic, Marko Arsenovic, Andras Anderla, Dubravko Culibrk, and Darko Stefanovic. Deep neural networks based recognition of plant diseases by leaf image classification. *Computational intelligence and neuroscience*, 2016, 2016.

[71] Chris Somerville, Heather Youngs, Caroline Taylor, Sarah C Davis, and Stephen P Long. Feedstocks for lignocellulosic biofuels. *science*, 329(5993):790–792, 2010.

[72] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[73] Ben Tan, Junping Zhang, and Liang Wang. Semi-supervised elastic net for pedestrian counting. *Pattern Recognition*, 44(10-11):2297–2304, 2011.

[74] A Taskinen, M Vilkko, P Itävuo, and Antti Jaatinen. Fast size distribution estimation of crushed rock aggregate using laserprofilometry. *IFAC Proceedings Volumes*, 44(1):12132–12137, 2011.

[75] Tensorflow team. Eager execution in tensorflow. https://www.tensorflow.org/guide/eager, 2019. Accessed: 2019-07-21.

[76] Mustafa Teke, Hüsne Seda Deveci, Onur Haliloğlu, Sevgi Zübeyde Gürbüz, and Ufuk Sakarya. A short survey of hyperspectral remote sensing applications in agriculture. In *2013 6th International Conference on Recent Advances in Space Technologies (RAST)*, pages 171–176. IEEE, 2013.

[77] Lisa Torrey and Jude Shavlik. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI Global, 2010.

[78] Teemu Väyrynen, Pekka Itävuo, Matti Vilkko, Antti Jaatinen, and Mika Peltonen. Mass-flow estimation in mineral-processing applications. *IFAC Proceedings Volumes*, 46(16):271–276, 2013.

[79] Ohmart Vega. Radiation-based weight systems for conveyors, 1 2013.

[80] Keith W Wendte, Andrey Skotnikov, and Kurian K Thomas. Sugar cane yield monitor, August 14 2001. US Patent 6,272,819.

[81] Marco Wiering and Martijn Van Otterlo. Reinforcement learning. *Adaptation, learning, and optimization*, 12:3, 2012.

[82] Hulya Yalcin and Salar Razavi. Plant classification using convolutional neural networks. In *2016 Fifth International Conference on Agro-Geoinformatics (Agro-Geoinformatics)*, pages 1–5. IEEE, 2016.

[83] Xuesong Yin, Songcan Chen, Enliang Hu, and Daoqiang Zhang. Semi-supervised clustering with metric learning: An adaptive kernel method. *Pattern Recognition*, 43(4):1320–1333, 2010.

[84] Wenli Zhang. *Experimental and computational analysis of random cylinder packings with applications*. PhD thesis, Louisiana State University, 2006.

[85] Zhi-Hua Zhou and Ming Li. Semi-supervised regression with co-training. In *IJCAI*, volume 5, pages 908–913, 2005.

[86] Xiaojin Zhu and Andrew B Goldberg. Kernel regression with order preferences. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 681. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[87] Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

[88] Xiaojin Jerry Zhu. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005.

[89] Jacek M Zurada. *Introduction to artificial neural systems*, volume 8. West publishing company St. Paul, 1992.